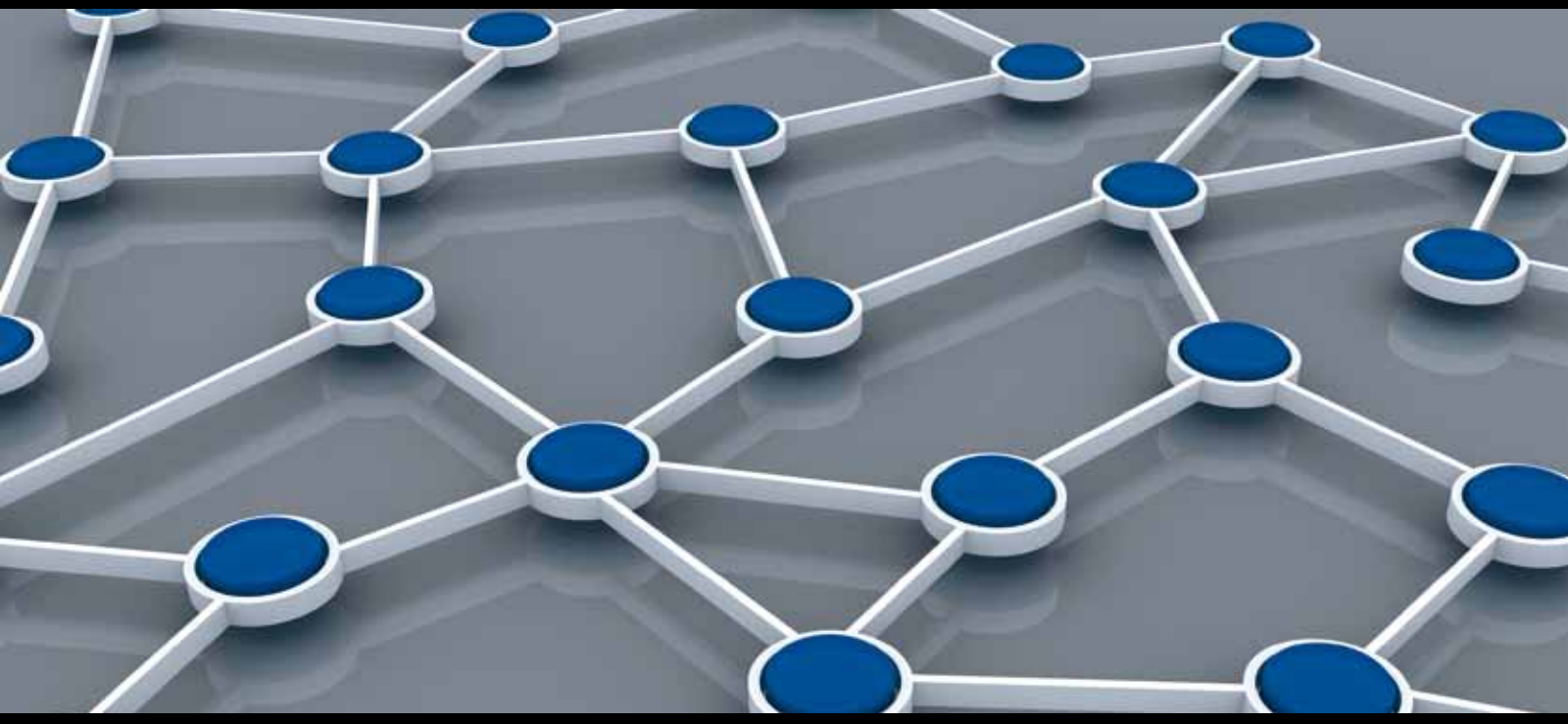
International Journal of Distributed Sensor Networks

# Deploying Real-Life WSN Applications: Challenges, Solutions, and Future Directions

5.2023

Guest Editors: Regina B. Araujo, Carlos H. C. Ribeiro, Damla Turgut, Jo Ueyama, and Torsten Braun



# Deploying Real-Life WSN Applications: Challenges, Solutions, and Future Directions

International Journal of Distributed Sensor Networks

# Deploying Real-Life WSN Applications: Challenges, Solutions, and Future Directions

Guest Editors: Regina B. Araujo, Carlos H. C. Ribeiro, Damla Turgut, Jo Ueyama, and Torsten Braun

# Editorial Board

# Contents

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 610892, 2 pages
http://dx.doi.org/10.1155/2013/610892

*Editorial*

# Deploying Real-Life WSN Applications: Challenges, Solutions, and Future Directions

**Regina B. Araujo,[1] Carlos H. C. Ribeiro,[2] Damla Turgut,[3] Jo Ueyama,[4] and Torsten Braun[5]**

[1] *Federal University of São Carlos, Brazil*
[2] *Technological Institute of Aeronautics, Brazil*
[3] *University of Central Florida, USA*
[4] *University of São Paulo, Brazil*
[5] *University of Bern, Imho, Switzerland*

Correspondence should be addressed to Regina B. Araujo; regina@dc.ufscar.br

Received 19 March 2013; Accepted 19 March 2013



The massive data from increasing numbers of Internet-enabled sensors and wireless sensor networks (WSNs), as well as humans as sensors, among other sources, is paving the way to an observable world in a nonprecedent scale. Great challenges lie ahead, not in the least for researchers worldwide, to collect and make sense of this information so that adaptable and smarter systems and applications can be made available. This International Journal of Distributed Sensor and Network special edition presents solutions for some of the challenges of design and implementation of WSN applications, involving quality of data, reliability, security, and limitation of resources regarding energy, storage, and processing of the collected data.

For WSN applications in a home environment, in which fixed and mobile sensors coexist, traditional WSN solutions do not meet requirements accordingly. One of our papers, "*Architecture and routing protocols for smart wireless home sensor networks*," by Yang Xu et al., presents a solution of architecture, routing protocol, and recovery mechanism to integrate fixed sensors with mobile sensors in a wireless home sensor network. AC-powered sensors are used as backbone nodes for data retransmission while battery-powered sensors are used as leaves to transmit only data that are relevant to them. A mechanism based on prime numbers is used to indicate the route towards the destination. A path recovery algorithm is presented to handle node mobility and backbone node failures.

Data quality insurance and reliability are important requirements for WSN applications. However, due to the unreliable nature of wireless communication and resource constraints in sensor nodes, these requirements pose challenges for WSN researchers. Two of the papers address such issues. In "*LiReTa: A lightweight reliable transmission scheme for wireless sensor networks using cross-layer information*," the authors, Gawon Lee, and Eui-Nam Huh, introduce a solution to reliability based on overhearing data transfer protocol, which uses "implicit acknowledgement" in a cross-layer design. In another paper, "*Node classification based on functionality in energy efficient and reliable wireless sensor networks*," by Ning Sun et al., a solution using different MAC and network strategies is presented to ensure reliability in data dissemination.

With the widespread deployment of WSN in different domains of application, from home to defense, WSN applications can be targets to different attacks. As an open transmission media, a sensor network can be subject to radio jamming attacks that can be challenging to defend, provoking corrupted packets and low network throughput. Authors Yanmin Zhu et al., present an adaptive solution to jamming signals attacks in the paper "*Optimal adaptive anti-jamming in wireless sensor networks*" by combining powerful existent techniques with a novel mechanism, based on a Markov decision process, which computes the best antijamming strategy under jamming conditions varying over time.

Given the sensor nodes limited resources for storage and processing, the integration of cloud computing and WSN can provide a powerful solution. In the paper entitled "*A survey on sensor-cloud: architecture, applications, and approaches*," by Atif M. AlAmri et al., the authors present a comprehensive survey of the recent works on sensor-cloud infrastructure including an overview of the sensor-cloud platform, its definition, architecture, and applications. The manuscript also highlights the research challenges and future research directions in this area.

## Acknowledgments

The editors of this issue would like to thank the reviewers for their valuable and positive comments and suggestions to improve the quality of the papers.

*Regina B. Araujo*
*Carlos H. C. Ribeiro*
*Damla Turgut*
*Jo Ueyama*
*Torsten Braun*

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 917923, 18 pages
http://dx.doi.org/10.1155/2013/917923

*Review Article*

# A Survey on Sensor-Cloud: Architecture, Applications, and Approaches

**Atif Alamri, Wasai Shadab Ansari, Mohammad Mehedi Hassan, M. Shamim Hossain, Abdulhameed Alelaiwi, and M. Anwar Hossain**

*College of Computer and Information Sciences, King Saud University, P.O. Box 51178, Riyadh 11543, Saudi Arabia*

Correspondence should be addressed to Atif Alamri; atif@ksu.edu.sa

Received 25 May 2012; Revised 30 September 2012; Accepted 28 November 2012

Academic Editor: Damla Turgut



Nowadays, wireless sensor network (WSN) applications have been used in several important areas, such as healthcare, military, critical infrastructure monitoring, environment monitoring, and manufacturing. However, due to the limitations of WSNs in terms of memory, energy, computation, communication, and scalability, efficient management of the large number of WSNs data in these areas is an important issue to deal with. There is a need for a powerful and scalable high-performance computing and massive storage infrastructure for real-time processing and storing of the WSN data as well as analysis (online and offline) of the processed information under context using inherently complex models to extract events of interest. In this scenario, cloud computing is becoming a promising technology to provide a flexible stack of massive computing, storage, and software services in a scalable and virtualized manner at low cost. Therefore, in recent years, Sensor-Cloud infrastructure is becoming popular that can provide an open, flexible, and reconfigurable platform for several monitoring and controlling applications. In this paper, we present a comprehensive study of representative works on Sensor-Cloud infrastructure, which will provide general readers an overview of the Sensor-Cloud platform including its definition, architecture, and applications. The research challenges, existing solutions, and approaches as well as future research directions are also discussed in this paper.

## 1. Introduction

The advancement and application of wireless sensor networks become an invincible trend into the various industrial, environmental, and commercial fields. A typical sensor network may consist of a number of sensor nodes acting upon together to monitor a region and fetch data about the surroundings. A WSN contains spatially distributed self-regulated sensors that can cooperatively monitor the environmental conditions, like sound, temperature, pressure, motion, vibration, pollution, and so forth [1, 2]. Each node in a sensor network is loaded with a radio transceiver or some other wireless communication device, a small microcontroller, and an energy source most often cells/battery. The nodes of sensor network have cooperative capabilities, which are usually deployed in a random manner. These sensor nodes basically consist of three parts: sensing, processing, and communicating [3]. Some of the most common sensor devices deployed in sensor network as sensor nodes are camera sensor, accelerometer sensor, thermal sensor, microphone sensor, and so forth.

Currently, WSNs are being utilized in several areas like healthcare, defense such as military target tracking and surveillance [4, 5], government and environmental services like natural disaster relief [6], hazardous environment exploration, and seismic sensing [7], and so forth. These sensors may provide various useful data when they are closely attached to each of their respective applications and services directly [8]. However, sensor networks have to face many issues and challenges regarding their communications (like short communication range, security and privacy, reliability, mobility, etc.) and resources (like power considerations, storage capacity, processing capabilities, bandwidth availability, etc.). Besides, WSN has its own resource and design constraints. Design constraints are application specific and dependent on monitored environment. Based on the monitored environment, network size in WSN varies. For

monitoring a small area, fewer nodes are required to form a network whereas the coverage of a very large area requires a huge number of sensor nodes. For monitoring large environment, there is limited communication between nodes due to obstructions into the environment, which in turn affects the overall network topology (or connectivity) [9]. All these limitations on sensor networks would probably impede the service performance and quality. In the midst of these issues, the emergence of cloud computing is seen as a remedy.

Cloud computing has been evolved as the future generation's computing paradigm. The US NIST (National Institute of Standards and Technology) defines the concept of Cloud computing as follows:

> *Cloud computing is a model for enabling convenient, on demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [5].*

Cloud computing allows the systems and users to use Platform as a Service (PaaS), for example, Operating Systems (OSs), Infrastructure as a Service (IaaS), for example, storages and servers, and Software as a Service (SaaS), for example, application level programs, and so forth at a very low cost which are being provided by several cloud providers (e.g., example Amazon, Google, and Microsoft) on the basis of pay per use services [10]. Cloud computing platform dynamically provisions, configures, and reconfigures the servers as and when needed by end users. These servers can be in the form of virtual machines or physical machines in the cloud. Cloud computing renders the two major trends in IT: (1) efficiency, which is achieved through the highly scalable hardware and software resources, and (2) agility, which is achieved through parallel batch processing, using computer-intensive business analytics and real-time mobile interactive applications that respond to user requirements [11]. The benefits of cloud computing are that the end users need not to worry about the exact location of servers and switch to their application by connecting to the server on cloud and start working without any hassle.

Sensor-Cloud infrastructure [12] has been evolved and proposed by several IT people in the present days. Sensor-Cloud infrastructure is the extended form of cloud computing to manage the sensors which are scattered throughout the network (WSN). Due to the increasing demand of sensor network applications and their support in cloud computing for a number of services, Sensor-Cloud service architecture is introduced as an integration of cloud computing into the WSN to innovate a number of other new services. When WSN is integrated with cloud computing environment, several shortfalls of WSN like storage capacity of the data collected on sensor nodes and processing of these data together would become much easier. Since cloud computing provides a vast storage capacity and processing capabilities, it enables collecting the huge amount of sensor data by linking the WSN and cloud through the gateways on both sides, that is, sensor gateway and cloud gateway. Sensor gateway collects information from the sensor nodes of WSN, compresses it, and transmits it back to the cloud gateway which in turn decompresses it and stores it in the cloud storage server, which is sufficiently large [13].

Sensor-Cloud can be used in many real-life applications like environmental monitoring, disaster monitoring, telemetric, agriculture, irrigation, healthcare, and so forth. As an illustration, we can use the Sensor-Cloud infrastructure for deploying health-related applications such as monitoring patients with cardiovascular disease, blood sugar followup, sleep activity pattern monitoring, diabetics monitoring, and so forth. In traditional approach, the trials of individual's data like level of blood sugar, weight, heart rate, pulse rate, and so forth are reported everyday through some telemedicine interface [14]. The patient's trial information is sent to a dedicated server and is stored there for doctors or caregivers to analyze it sometime later. This system suffers from a level of adversity when the patient randomly moves from its current location, that is, when a patient is "on the go." Thus, a more progressive, rapid, and mobile approach is needed where the recorded data from several sensor nodes of a WSN can be processed in pipelined and parallel fashion, and thereby to make the system easier to scale and be cost-effective in terms of resources available. The pipeline processing of data sets or instructions enable the overlapped operations into a conceptual pipe with all the stages of pipes processing simultaneously but handling of the sensor data stream is not that straight forward and will be dependent on the nature of the algorithm [15]. The integration of Sensor-Cloud can serve as a remedy in this direction.

This paper presents a comprehensive survey on Sensor-Cloud. Section 2 of this paper provides a brief overview of this concept, including its definition, architecture, and its advantage. Section 3 discusses the service life cycle model and layered approach of Sensor-Cloud architecture. Section 4 provides the Sensor-Cloud service creation and renovation capabilities. Section 5 presents the multiservice provisioning on multiple platforms. Section 6 discusses the technical comparison of different approaches in sensor-cloud infrastructure. Next, Section 7 presents several issues that may arise in Sensor-Cloud and some approaches to address these issues. Finally, we summarize and conclude the survey in Section 8.

## 2. Overview and Related Works

Sensor-Cloud is a new paradigm for cloud computing that uses the physical sensors to accumulate its data and transmit all sensor data into a cloud computing infrastructure. Sensor-Cloud handles sensor data efficiently, which is used for many monitoring applications.

*2.1. What is a Sensor-Cloud?* According to IntelliSys, Sensor-Cloud can be defined as follows:

> *An infrastructure that allows truly pervasive computation using sensors as an interface between physical and cyber worlds, the data-compute clusters as the cyber backbone and the internet as the communication medium [16, 17].*

According to MicroStrains's Sensor-Cloud definition "*it is a unique sensor data storage, visualization and remote management platform that leverage [sic] powerful cloud computing technologies to provide excellent data scalability, rapid visualization, and user programmable analysis. It is originally designed to support long-term deployments of MicroStrain wireless sensors, Sensor-Cloud now supports any web-connected third party device, sensor, or sensor network through a simple OpenData API*" [18].

A Sensor-Cloud collects and processes information from several sensor networks, enables information sharing on big scale, and collaborates with the applications on cloud among users. It integrates several networks with a number of sensing applications and cloud computing platform by allowing applications to be cross-disciplinary that may span over multiple organizations [17]. Sensor-Cloud enables users to easily gather, access, process, visualize, analyze, store, share, and search for a large number of sensor data from several types of applications and by using the computational IT and storage resources of the cloud.

In a sensor network, the sensors are utilized by their specific application for a special purpose, and this application handles both the sensor data and the sensor itself such that other applications cannot use this. This makes wastage of valuable sensor resources that may be effectively utilized when integrating with other application's infrastructure. To realize this scenario, Sensor-Cloud infrastructure is used that enables the sensors to be utilized on an IT infrastructure by virtualizing the physical sensor on a cloud computing platform. These virtualized sensors on a cloud computing platform are dynamic in nature and hence facilitate automatic provisioning of its services as and when required by users [19]. Furthermore, users need not to worry about the physical locations of multiple physical sensors and the gapping between physical sensors; instead, they can supervise these virtual sensors using some standard functions [12].

Within the Sensor-Cloud infrastructure, to obtain QoS, the virtual sensors are monitored regularly so users can destroy their virtual sensors when they becomes meaningless [20]. A user interface is provisioned by this Sensor-Cloud infrastructure for administering, that is, for controlling or monitoring the virtual sensors, provisioning and destroying virtual sensors, registering and deleting of physical sensors, and for admitting the deleting users. For example, in a health monitoring environment, a patient may use a wearable computing system (that may include wearable accelerometer sensors, proximity sensors, temperature sensors, etc.) like Life Shirt [21] and Smart Shirt [22] or may use a handheld device loaded with sensors, and consequently the data captured by the sensors may be made accessible to the doctors. But out of these computing systems, active continuous monitoring is most demanding, and it involves the patient wearing monitoring devices to obtain pervasive coverage without being inputted or intervened [23]. However, the diverse monitoring scheme defers in their QoS requirements, which are as follows.

(1) *Patient-Centric Healthcare-QoS* refers to monitoring delay and reliability of message delivery.

(2) *Network-Centric Healthcare-QoS* refers to a number of patients supported and message throughput.

(3) *Healthcare professional-centric Healthcare-QoS* refers to cognitive load of healthcare professionals and the number of correct medical decisions.

Sensor modeling language (SML) [24] can be used to represent any physical sensor's metadata like their type, accuracy, their physical location, and so forth. It also uses XML encoding for the measurement and description processes of physical sensors. This XML encoding for physical sensors enabled these to be implemented across several different hardware, platforms (OS), applications, and so forth with relatively less human intervention. To transliterate the commands coming from users to virtual sensors and in turn to the commands for their pertinent physical sensors, a mapping is done between the physical and virtual sensors.

*2.2. Architecture of Sensor-Cloud.* Cloud computing service framework delivers the services of shared network through which the users are benefited by the services, and they are not concerned with the implementation details of the services provided to them. When a user requests, the service instances (e.g., virtual sensors) generated by cloud computing services are automatically provisioned to them [12, 19].

Some previous studies on physical sensors focused on routing [25], clock synchronization [26], data processing [27], power management [28], OS [29], localization [30], and programming [31]. However, few studies concentrate on physical sensor management because these physical sensors are bound closely to their specific application as well as to its tangible users directly. However, users, other than their relevant sensor services, cannot use these physical sensors directly when needed. Therefore, these physical sensors should be supervised by some special sensor-management schemes. The Sensor-Cloud infrastructure would subsidize the sensor system management, which ensures that the data-management usability of sensor resources would be fairly improved.

There exists no application that can make use of every kind of physical sensors at all times; instead, each application required pertinent physical sensors for its fulfillment. To realize this concept, publish/subscription [32] mechanism is being employed for choosing the appropriate physical sensor [33]. In multiple sensor networks, every sensor network publishes its sensor data and metadata. The metadata comprises of the types, locations, and so forth for the physical sensors. Application either subscribes to one or maybe to more sensor networks to retrieve real-time data from the physical sensors by allowing each application to opt for the appropriate physical sensors' type. The Sensor-Cloud infrastructure procreates virtual sensors from multiple physical sensors, which can then be utilized by users. However, prior to availing the virtual sensor facility, users should probe first for the physical sensors' availability and might also inspect the physical sensors' faults to maintain the data quality emerging from these physical sensors. Also on every sensor node, application program senses the application and sends the sensor data back to the gateway in the cloud directly through

the base station or in multihop manner through other nodes [5, 13].

Sensor-Cloud infrastructure provides service instances (virtual sensors) automatically to the end users as and when requested, in such a way that these virtual sensors are part of their IT resources (like disk storage, CPU, memory, etc.) [34]. These service instances and their associated appropriate sensor data can be used by the end users via a user interface through the web crawlers as described in Figure 1. However, for the service instance generation, the IT resources (like CPU, storage devices, etc.), sensor capable devices, and service templates (which is used to create virtual sensors) should be prepared first in Sensor-Cloud infrastructure. Users make the request for service instances according to their needs by selecting an appropriate service template of Sensor-Cloud, which will then provide the needed service instances freely and automatically because of cloud computing services integration. Once service instances become useless, they can then be deleted quickly by users to avoid the utilization charges for these resources. Sensor service provider will manage the service templates (ST) and it can add or delete the new service template when the required template is no longer needed by applications and services [12]. Automation of services played a vital role in provisioning of cloud computing services, and automation can cause the delivery time of services to be better [19]. Before the emergence of cloud computing, services were provided by human influence and the performance metrics like efficiency, flexibility, delivery times, and so forth would have experienced an adverse effect on the system. However, the cloud computing service model has reduced the cost expenses and delivery time and has also improved the efficiency and flexibility.

The physical sensors are ranked on a basis of their sensor readings as well as on their actual distance from an event. The authors of [35] proposed a technique (FIND) to locate physical sensors having data faults by assuming a mismatch between the distance rank and sensor data rank. However, the study led by FIND aims at the assessment of physical sensors faults, and there is a close relation between the virtual and physical sensors and hence a virtual sensor will provide incorrect results if their relevant physical sensors are faulty. It concludes that the virtual and physical sensors cooperate while delivering the sensor data report to its applications, and to maintain the best report both the virtual and physical sensors should be faultless.

Since the cloud computing enables the physical sensors to be virtualized, the users of the Sensor-Cloud infrastructure need not to worry about the status of their connected physical sensors (i.e., whether a fault free or not). However, they should concern only with the status of their virtual sensors provided only when the users are not concerned with the accurate results. To achieve accurate results, users must be concerned about the status of physical sensors too. In a Sensor-Cloud infrastructure, sensors owners are free to register or unregister their physical sensors and can join this infrastructure. These IT resources (physical sensors, database servers, processors, etc.) and sensor devices are then prepared to become operational. After that, templates are created for generating the service instances (virtual sensors) and its groups (virtual sensors). Once templates are prepared, the virtual sensors are able to share the related and contiguous physical sensors to receive quality sensor data. Users then request these virtual sensors by choosing the appropriate service templates, use their service instances (virtual sensors) after being provisioned, and discharge them when became useless [12].

*2.3. Advantages of Sensor-Cloud.* Cloud computing is very encouraging solution for Sensor-Cloud infrastructure due to several reasons like the agility, reliability, portability, real-time, flexibility, and so forth. Structural health and environment-based monitoring contains highly sensitive data and applications of these types cannot be handled by normal data tools available in terms of data scalability, performance, programmability, or accessibility. So a better infrastructure is needed that may contain tools to cope with these highly sensitive applications in real time. In the following, we describe the several advantages and benefits of Sensor-Cloud infrastructure that may be the cause of its glory, and these are as follows.

(1) *Analysis.* The integration of huge accumulated sensor data from several sensor networks and the cloud computing model make it attractive for various kinds of analyses required by users through provisioning of the scalable processing power [34].

(2) *Scalability.* Sensor-Cloud enables the earlier sensor networks to scale on very large size because of the large routing architecture of cloud [16]. It means that as the need for resources increases, organizations can scale or add the extra services from cloud computing vendors without having to invest heavily for these additional hardware resources [36].

(3) *Collaboration.* Sensor-Cloud enables the huge sensor data to be shared by different groups of consumers through collaboration of various physical sensor networks [16]. It eases the collaboration among several users and applications for huge data sharing on the cloud.

(4) *Visualization.* Sensor-Cloud platform provide a visualization API to be used for representing the diagrams with the stored and retrieved sensor data from several device assets. Through the visualization tools, users can predict the possible future trends that have to be incurred [37].

(5) *Free Provisioning of Increased Data storage and Processing Power.* It provides free data storage and organizations may put their data rather than putting onto private computer systems without hassle. It provides enormous processing facility and storage resources to handle data of large-scale applications [16, 38].

(6) *Dynamic Provisioning of Services.* Users of Sensor-Cloud can access their relevant information from wherever they want and whenever they need rather than being stick to their desks [38].

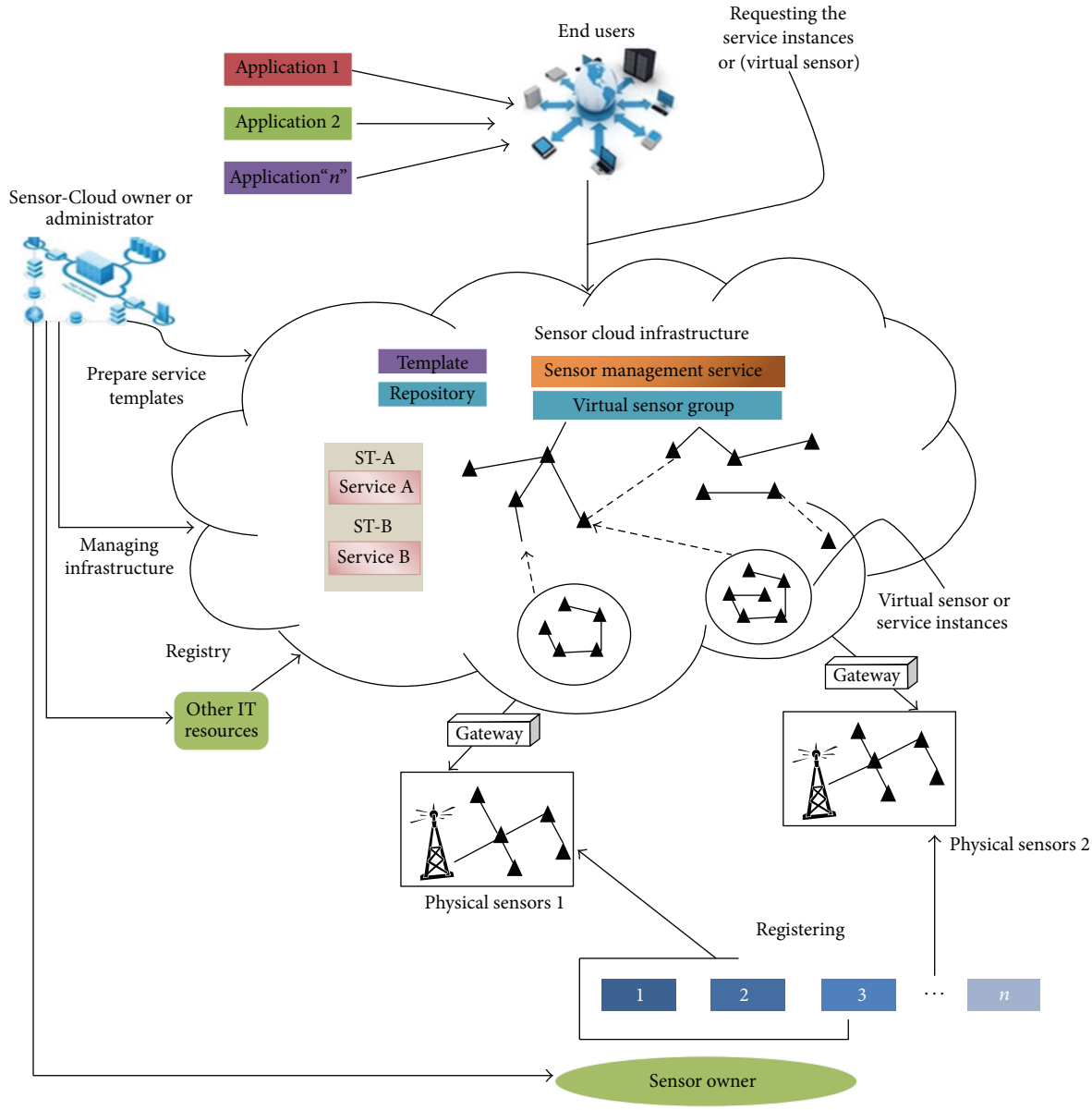



Figure 1: Brief overview of Sensor-Cloud architecture.

(7) *Multitenancy.* The number of services from several service providers can be integrated easily through cloud and Internet for numerous service innovations to meet user's demand [17]. Sensor-Cloud allows the accessibility to several numbers of data centers placed anywhere on the network world [36].

(8) *Automation.* Automation played a vital role in provisioning of Sensor-Cloud computing services. Automation of services improved the delivery time to a great extent [19].

(9) *Flexibility.* Sensor-Cloud provides more flexibility to its users than the past computing methods. It provides flexibility to use random applications in any number of times and allows sharing of sensor resources under flexible usage environment [16].

(10) *Agility of Services.* Sensor-Cloud provides agile services and the users can provision the expensive technological infrastructure resources with less cost [37]. The integration of wireless sensor networks with cloud allows the high-speed processing of data using immense processing capability of cloud.

(11) *Resource Optimization.* Sensor-Cloud infrastructure enables the resource optimization by allowing the sharing of resources for several number of applications [37]. The integration of sensors with cloud enables gradual reduction of resource cost and achieves higher gains of services. With Sensor-Cloud, both the small and midsized organizations can benefit from an enormous resource infrastructure without having to involve and administer it directly [36].

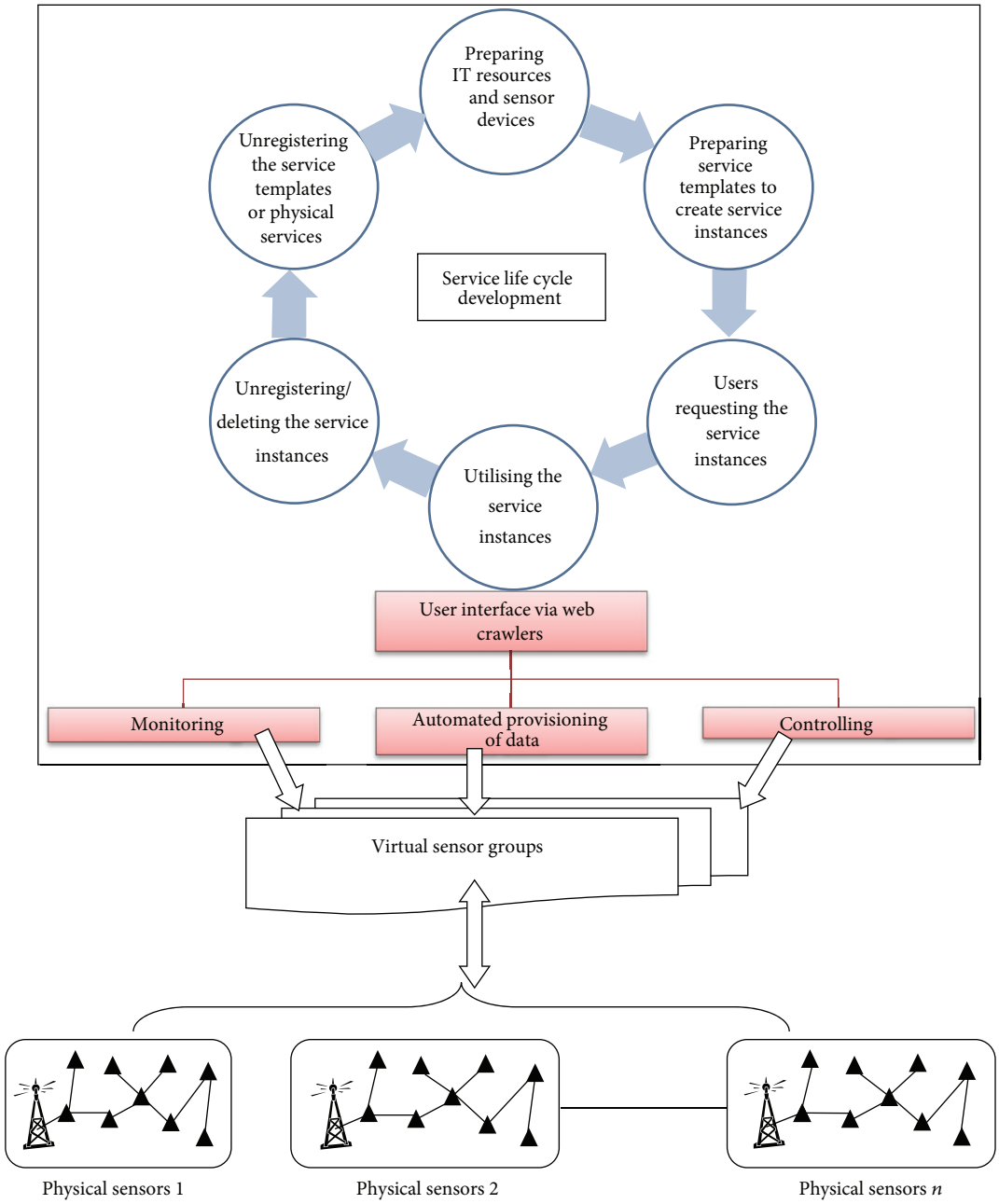



Figure 2: Demonstration of Sensor-Cloud life cycle development phases.

(12) *Quick Response Time.* The integration of WSN's with cloud provides a very quick response to the user, that is, in real-time due to the large routing architecture of cloud [39]. The quick response time of data feeds from several sensor networks or devices allows users to make critical decisions in near real time.

## 3. Sensor-Cloud Service Life-Cycle Model and Its Layered Structure

The service life-cycle model and the layered structure of Sensor-Cloud infrastructures are illustrated in the following.

*3.1. Service Life Cycle Model of Sensor-Cloud.* Before creating the service instances within Sensor-Cloud infrastructure, preparation phase [12] is needed, and this includes the following.

(1) Preparing the IT resources (processors, storage, disk, memory etc.).

(2) Preparing the physical sensor devices.

(3) Preparing the service templates.

Figure 2 depicts the Sensor-Cloud service life cycle. The users of the sensors can select the appropriate service template and request the required service instances. These

service instances are provided automatically and freely to the users, which can then be deleted quickly when they become useless. From a single service template, multiple numbers of service instances can be created. Service provider regulates the service templates and can add new service templates as and when required by a different number of users [8].

*3.2. Layered Structure of Sensor-Cloud.* Figure 3 depicts the layered architecture of the Sensor-Cloud platform, which is divided mainly into three layers:

(1) user and application layers,

(2) Sensor-Cloud and virtualization layers,

(3) template creation and tangible sensors layers [8].

*Layer 1.* This layer deals with the users and their relevant applications. Several users want to access the valuable sensor data from different OS platforms, such as mobile phones OS, Windows OS, or Mac OS for a variety of applications. This structure allows users of different platforms to access and utilize the sensor data without facing any problem because of the high availability of cloud infrastructure and storage.

*Layer 2.* This layer deals with virtualization of the physical sensors and resources in the cloud. The virtualization enables the provisioning of cloud-based sensor services and other IT resources remotely to the end-user without being worried about the sensors exact locations. The virtualized sensors are created by using the service templates automatically. Service templates are prepared by the service providers as service catalog, and this catalog enables the creation of service instances automatically that are accessed by multiple users [8, 19].

*Layer 3.* This is the last layer which deals with the service template creation and service catalog definition layers in forming catalog menu. Physical sensors are located and retrieved from this layer. Since each physical sensor has its own control and data collection mechanism, standard mechanisms are defined and used to access sensors without concerning the differences among various physical sensors [12]. Standard functions are defined to access the virtual sensors by the users. In Layer 2, Sensor-Cloud infrastructure translates standard functions of virtual sensors into some specific functions for diverse physical sensors in Layer 3. Physical sensors are XML encoded that enable the services provided through these sensors to be utilized on various platforms without being worried to convert them onto several platforms [8].

Sensor-Cloud provides a web-based aggregation platform for sensor data that is flexible enough to help in developing user-based applications. It allows users quick development and deployment of data processing applications and gives programming language flexibility in accordance to their needs [18].

## 4. Service Creation and Innovation Capabilities

Sensors are very limited and specific to their applications or services when they are linked to a typical sensor network. Therefore, the numbers of organizations that can provide the sensor services are very limited. However, when the services of sensors move onto the cloud, it is possible to include them to realize a variety of applications [8, 12]. A number of services can be provided to the users for different applications such as health applications, environmental monitoring, industrial tasks (e.g., refining), surveillance, senior residents monitoring, or even the applications that monitor the vibration in buildings during an earthquake.

In the Sensor-Cloud infrastructure, the sensors and service templates are constructed as catalog menu service on the cloud, and the requesters can create new sensor services with the existing sensors in these service instances. For example, service requester can create a sensor service to analyze the impact of earthquake to each floor or room of the rehabilitation center or hospital, and at the same time it can also create sensor services to support older residents with the same set of sensors (virtualized sensors). This service will then help the caregiver to shift the older adults one by one into a safe place. Using the identical sensor services for healthcare, another service requester can create dissimilar sensor service to track the patient's medicine intake and then to analyze the effectiveness of pills through the use of some selected healthcare sensors. Thus, the service requesters can be provided with new services using the same set of sensors on cloud service platforms. This will reduce the cost for resource usage and could have numerous elastic merits to it. In this section, several existing Sensor-Cloud applications are described.

*4.1. Existing Sensor-Cloud Applications.* There exist a number of services based on Sensor-Cloud infrastructure to store and process the sensor-based information. Few of them are described briefly as below.

*4.1.1. Nimbits.* Nimbit [40] is a free and social service that is used to record and share sensor data on cloud. It is a cloud-based data processing service and is an open-source platform for the IoT (Internet of Things). We can feed the versatile numeric, text-based, JSON, GPS, or XML values by creating a data point in the cloud. The data points can be connected to Scalable Vector Graphic (SVG) process control, spreadsheets, diagrams, websites, and more. Data points can also be configured to generate alerts data-relay to social networks and to perform calculations. Nimbits also provide an alert management mechanism, data compression mechanism, and data calculation on received sensor data by employing some simple mathematical formulas.

*4.1.2. Pachube Platform.* Pachube [41] is one of the first online database service providers, which allows us to connect sensor data to the web. It is a real-time cloud-based platform for IoT with a scalable infrastructure that enables us to configure

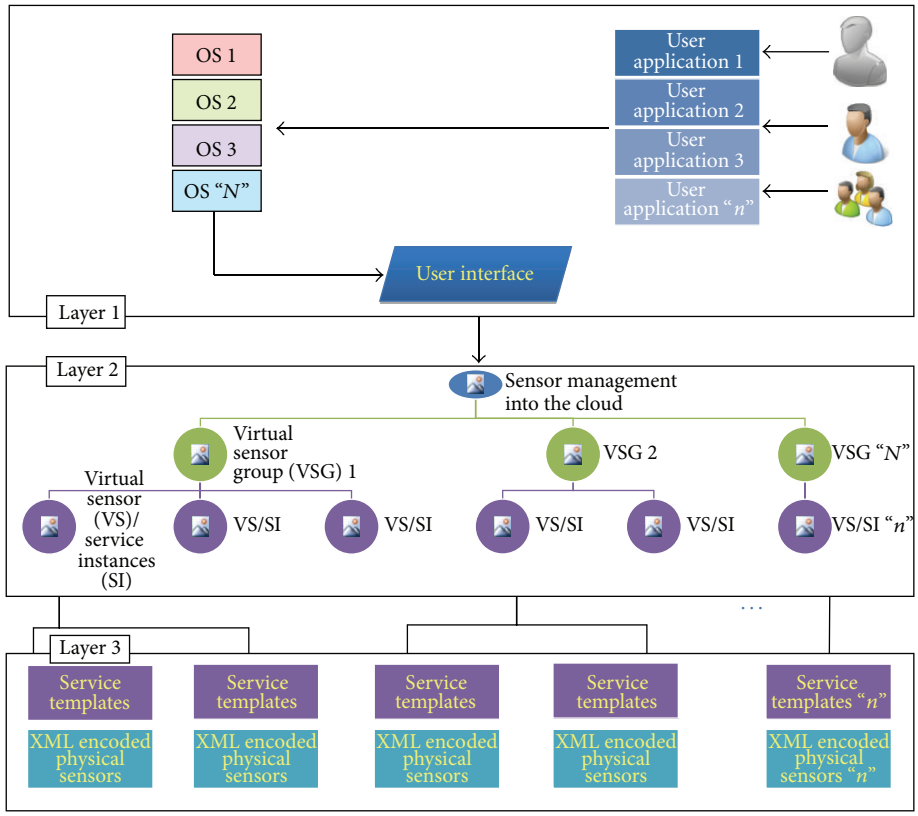



Figure 3: Layered structure of Sensor-Cloud model.

IoT products and services, store, share, and discover real-time energy, environment, and healthcare sensor data from devices and buildings around us. Pachube has a very interactive website for managing the sensor data and an open easily-accessible API. Pachube system provides free usage and has several numbers of interfaces for producing a sensor or mobile-based applications for managing the sensor data within a cloud framework anytime.

*4.1.3. IDigi.* iDigi [42] is a machine-to-machine (M2M) platform as a service PaaS that minimizes the barriers to build scalable, secure, and cost-effective solutions, which can bind the enterprise applications and device assets together. iDigi eases the connectivity of remote assets devices and provides all the tools to manage, store, connect, and move the information across the enterprise irrespective of its reache. To simplify the remote device connectivity and integration, it uses connector software called iDigi Dia. Regardless of the network location, iDigi platform manages communication between remote device assets and enterprise applications.

*4.1.4. ThingSpeak.* ThingSpeak [43] is another open source IoT application and has an open API to store and retrieve data from device assets or things via LAN or using HTTP over the Internet. With this platform, location tracking applications, sensor logging applications, and social network of device assets with proper update of its status can be created. ThingSpeak allows numeric data processing like averaging, time-scaling, rounding, median, summing, and so forth to store and retrieve the numeric and alphanumeric data. ThingSpeak application features a JavaScript-based charts, read/write API key management, and a time-zone management.

Although the above services are able to visualize the sensor data and sensor-driven information, they are lacking secure access to data and interface availability for linking the external or mobile applications for further processing. It means that most of these aforementioned projects do not address the issues of data management and interoperability issues caused by heterogeneous data resources found in the present modern environmental tracking or electronic healthcare systems. But introducing these aforementioned works with Cloud computing infrastructure may overcome the issues related to heterogeneous data access and data management functionality [37].

*4.2. Emerging Sensor-Cloud Applications.* There are many other applications that are emerging based on the Sensor-Cloud infrastructure, which can be summarized as follows.

*4.2.1. Ubiquitous Healthcare Monitoring.* Sensor-Clouds can be used for health monitoring by using a number of easily available and most often wearable sensors like accelerometer sensors, proximity, ambient light and temperature sensors,

and so forth to collect patient's health-related data for tracking sleep activity pattern, blood sugar, body temperature, and other respiratory conditions [44]. These wearable sensor devices must have support of BWI (Bluetooth's wireless interface), UWB (Ultra wideband), and so forth interface for streaming of data and are connected wirelessly to any smartphone through this interface. These smart phone devices pretend to function like a gateway between the remote server and sensor through the Internet, maybe GPRS/Wi-Fi, or other sort of gateways.

To transform this system into services-based structure, web-services-based interfaces are used by smart phone device to connect to the server [45]. The system prototype should have made to be robust, mobile, and scalable. Robust in the sense means that it should recover itself from circumstances, which may lack connectivity issues due to power (i.e., battery), failure, or gateway cutoff to patient's wearable devices [46]. Mobile in the sense means that it should be capable of tracking signals into heterogeneous environments; that is, it must catch the signals irrespective of whether the patient went outside or still resided into the hospital/building. It should be scalable so that it could be deployed easily for several users concurrently without affecting the performance metrics.

Finally, such prototype system should be retargetable and extensible in nature. Retargetable refers to the fact that it can handle various displays with distinct form factors and screen resolution. It means that the same health applications can be displayed to any smartphone display like PDA (personal digital assistant) or to a bigger console device in a hospital where doctors, helpers, or nurses may track the acquired data or processed information from distance. The extensibility aspect requires that if any newer sensing devices are introduced into the system for acquiring the patient's health-based information, the system should function efficiently and conveniently without affecting backend server of the services [47]. In this platform, context awareness can be achieved that can direct us to derive a better level of emergency services to the patient [20]. The information regarding recent operational laboratories, missing doses of pills, number of handicaps, and other situations would be helpful in health monitoring. The system should not adhere to any changes made into the operating system or intermediate components of sensing devices and is designed in such a way that it would cause minimal disturbance to services provided to existing end users of the system [15]. The whole scenario for monitoring the healthcare department sensors is shown in Figure 4.

In this scenario, several numbers of sensors pick up the patient data, and these accumulated data are uploaded to a server on cloud. If any noise data is found, they are filtered using some filtering mechanism on a server. The doctors/health employees, nurses, and others can then access the patients' data on cloud through a web service portal after being authenticated/permitted by the patient.

*4.2.2. Environmental Monitoring for Emergency/Disaster Detection.* In environmental applications, it is possible to detect the earthquake and volcano explosion before its eruption by continuously monitoring them through the use of several numbers of different sensors like strain, temperature, light, image, sound, acceleration, barometer sensors, and so forth through the use of wireless sensor networks [48]. Through the Sensor-Cloud infrastructure, the sensor instances engaged in environmental monitoring can be used in parallel with several other sensor instances, for example, by the healthcare department to avoid any future casualty, or with crop harvesting application services to avoid the damage caused by bad weather condition.

*4.2.3. Telematics.* Sensor-Clouds can be used for telematics, meant to deploy the long distance transmission of our computerized or information to a system in continuum. It enables the smooth communication between system and devices without any intervention [17].

*4.2.4. Google Health.* It is a centralization service of Google that provides personal health information [49] and serves as cloud health data storages. Google users are allowed to monitor their health records by logging into their accounts at collaborated cloud health service providers into the Google health system. However, in a recent declaration Google has announced the discontinuation of this health service.

*4.2.5. Microsoft HealthVault.* This cloud platform is developed by Microsoft to store and maintain health and fitness-related information [50]. HealthVault helps users to store, gather, and share their health relevant information and its data can be acquired from several pharmacies, cloud providers, health employees, health labs, equipment, and from the users itself.

*4.2.6. Agriculture and Irrigation Control (Field Server Sensors).* Sensor-Cloud can be used in the field of agriculture to monitor the crop fields in order to upkeep it. For this, a field server is developed that comprises of a camera sensors, air sensor, temperature sensor, $CO_2$ concentration sensor, soil moisture and temperature sensors, and so forth. These sensors continuously upload the field data via Wi-Fi access point to the field owner to track the health of their crops [51]. This can also be used for harvesting.

*4.2.7. Earth Observation.* A sensor grid is developed for data gathering from several GPS stations, to process, analyze, manage, and visualize the GPS data [52]. This GPS data would then be uploaded onto the cloud for efficient monitoring, early warning, and decision-making capability for critical situations like the volcanic eruptions, earthquakes, tsunamis, cyclones, and so forth to the users all around the world.

*4.2.8. Transportation and Vehicular Traffic Applications.* Sensor-Cloud can be used to provide an efficient, stable, equilibrium, and sustainable tracking system. Earlier existing technologies like GPS navigation can only track the status and current location of vehicle. On the other hand, when vehicle monitoring is implemented using cloud computing,

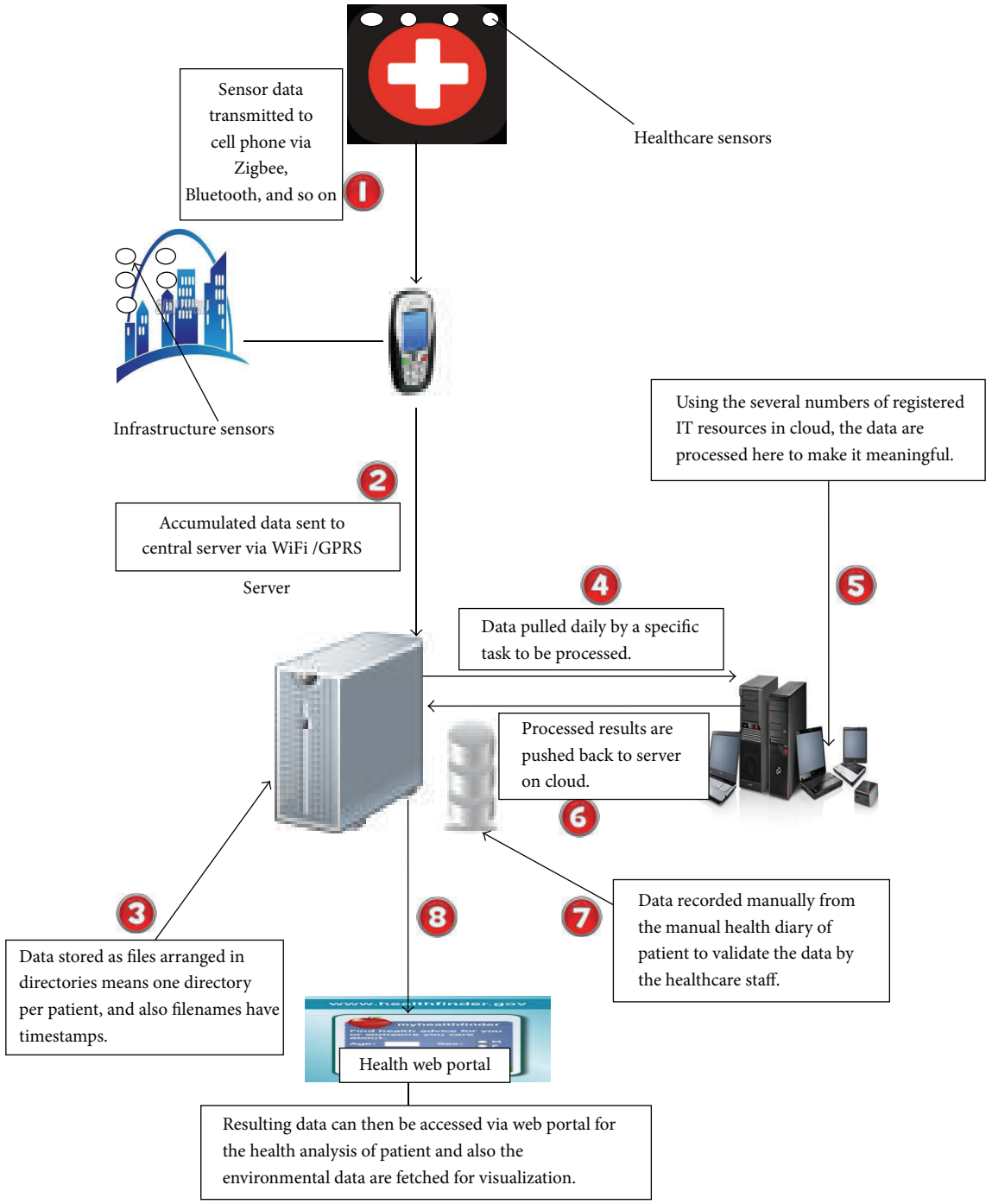



Figure 4: Sensor-Cloud implementation in healthcare.

it is possible to incorporate centralized web service, GPS and GSM enabled devices, and embedded device with sensors [36], which will provide the following benefits:

(i) to identify the current name of the location,

(ii) to predict the time of arrival,

(iii) to find status of driver via alcohol breath sensor,

(iv) to find the total distance covered,

(v) to track the level of fuel.

All the data fetched are stored onto some centralized server that will be resided into the cloud. The vehicle owner can access this data on cloud via web portal and can retrieve all data on cloud in real time to visualize the vehicle information.

*4.2.9. Tunnel Monitoring.* WSN can be used to implement the distributed sensing of light levels inside the tunnel and underbridges to provide necessary input information for adapting light functionality. This tunnel information can be put onto the cloud and is used to monitor the light intensity in real time to avoid the automobile users (drivers) casualty and to save the energy spent unnecessarily for lightening throughout the day [53].

*4.2.10. Wildlife Monitoring.* Sensor-Cloud can also be used for tracking the wildlife sanctuaries, forests, and so forth to regularly monitor the endangered species in real time.

## 5. Multiservice Provisioning on Multiple Platforms

Integrating the WSN into heterogeneous networks is a typical task. The reason behind this is the absence of standardized data exchange functions, which may support the participating subnetworks of heterogeneous network. Using XML in sensor networks encourages the interchangeability of different types of sensors and systems. The lower part of Figure 5 deals with the XML [24] encoded physical sensors. The XML encoding defines some set of rules for these physical sensors such that it will be both human readable and machine readable with less intervention and will enable these to be implemented on several numbers of different platforms. XML enables documents to give physical sensor's metadata, that is, the type of the physical sensors, its specifications, the accuracy or intensity of these physical sensors, the exact location, and so forth. But sensor nodes have limited storage and power constraints, and conflict may occur while using the XML encoding. For this reason, the XML support should be based on efficient data binding techniques to preserve the time, space, and energy by minimizing the XML overhead [34]. This whole scenario can be depicted by Figure 5.

The author in [54] has proposed to access the sensor information by using the Web Service Description Language (WSDL) and structure data, so that the multiple applications may access sensor information. But the key issue in using the web services on sensor nodes is energy and bandwidth overhead of structured data formats used in the web services [34]. In heterogeneous sensor networks, integration is a complex task because there is an absence of standardized data exchange format between the heterogeneous systems and networks. XML has evolved to overcome this insufficiency by providing a standard data exchange format between heterogeneous network and systems. Because of the limited hardware resources within sensor networks, XML usage was not fully introduced earlier. But now XML usage in sensor networks is made applicable by introducing the XML template objects in an optimized manner [55]. XML is basically a key feature towards the service-oriented sensor networks and a proper medium to support complex data management and heterogeneous sensor networks.

To enable the applications to communicate with each other and to provide remote access to the services offered by Sensor-Cloud platform, web services are introduced [37]. Web services mainly refers to access the services over Internet connection. It has WSDL (Web Service Description Language) definitions, which describe what the web service can do, how a web service can be used by client applications, and where the web service is located. SOAP messages are used to communicate with web services, and these SOAP messages are XML based that are transported over the Internet protocols like SMTP, FTP, and HTTP.

## 6. Comparison of Different Approaches in Sensor-Cloud Infrastructure

In this section, we first present the advantages and disadvantages of Sensor-Cloud infrastructure in terms of agility, reliability, portability, real-time, and flexibility. Next, we provide a technical comparison of different messaging approaches and algorithms used in several existing research on Sensor-Cloud.

*Pros of Sensor-Cloud Infrastructure*:

(i) Service requesters or end users can control the service instances freely [12].

(ii) End users can inspect the status of their relevant virtual sensors.

(iii) Service requesters can use the virtual sensors without worrying about the implementations detail [12].

(iv) The client/users need not to worry about the exact locations and detailed description of their sensors [45].

(v) The service instances are automatically provisioned whenever a request is made [19].

(vi) The IT resources and sensors are released as and when the required job is over, which means that users can delete them when they become nonuseful.

(vii) Usage of physical sensors can be tracked by the sensor owner.

(viii) Sensor data are available all the time for a number of various applications until the connection is provided [12].

(ix) The Sensor-Cloud architecture provides an extensible, open, interoperable, and intelligent sensor network for service provisioning in health care [56].

(x) The cost of IT resources and WSN infrastructure is reduced when integrating with Internet/Cloud [57].

(xi) End users can also create the sensors group dynamically in the form of virtual-sensor groups to innovate the new services [12].

Besides these advantages, the Sensor-Cloud infrastructure also has some drawbacks and these are as follow.

*Cons of Sensor-Cloud Infrastructure*:

(i) The IT resources and physical sensors should be prepared prior to operation of the Sensor-Cloud infrastructure [8].

(ii) The Sensor-Cloud infrastructure will not provide much accurate data as in the case of direct sharing of physical sensors data [12].

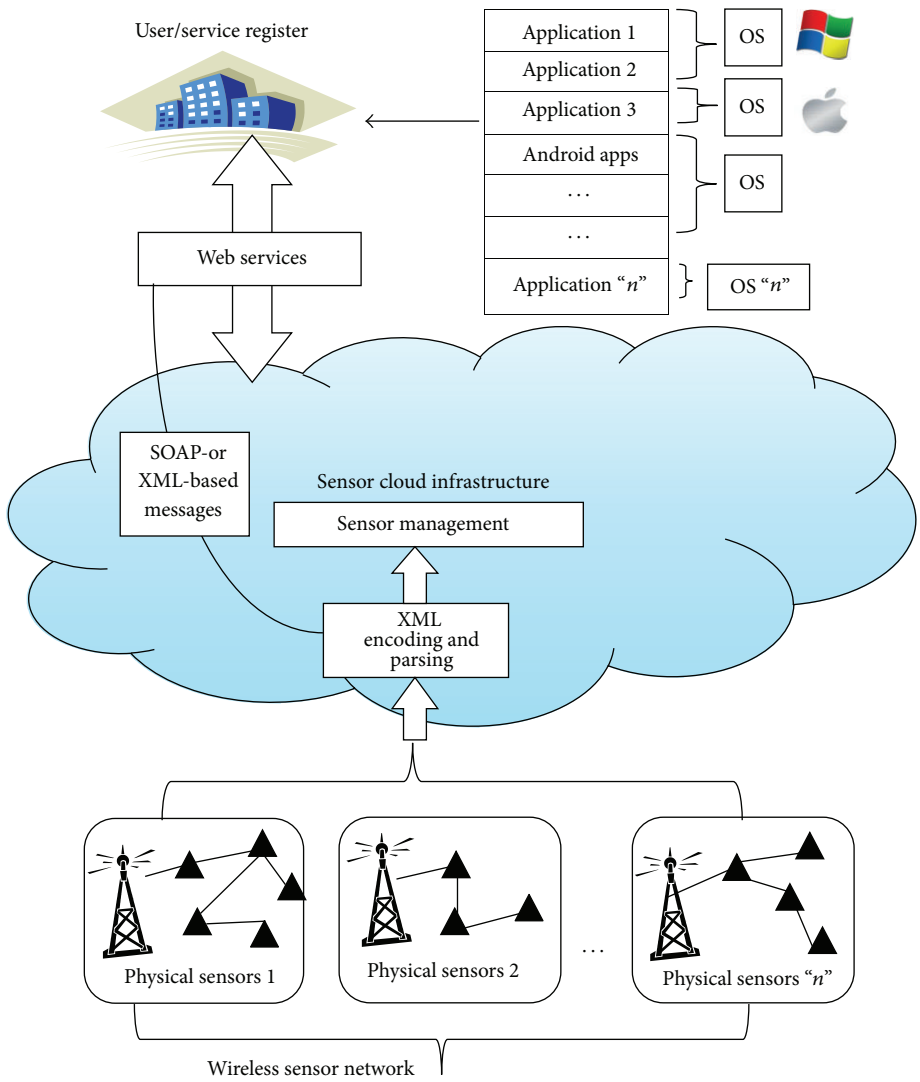



Figure 5: XML encoded physical sensors into the Sensor-Cloud.

(iii) Sensor-Cloud infrastructure is vulnerable and more prone to sophisticated distributed intrusion attacks like DDOS (distribute denial of service) and XSS (cross-site scripting) [58].

(iv) A continuous data connectivity is needed between end users and Sensor-Cloud server [15].

Because of continuous processing and wireless transmission, power is another factor that has to be dealt with carefully in CSI [45].

We now highlight the differences among the existing works on Sensor-Cloud in terms of message passing protocols, data caching, QoS routing, processing, event matching, and other algorithms, as shown in Table 1.

## 7. Issues and Challenges While Designing Sensor-Cloud Infrastructure in Several Applications

There are several issues like designing, engineering, reliable connection, continuous data flow, power issues, and so forth that need to be handled while proposing Sensor-Cloud infrastructure for health care and other different applications [68]. Some of the main issues are as follows.

*7.1. Design Issues.* There are several issues while designing the system in real scenario like nursing home, health care, hospitals, and so forth, which require *fault-tolerant* and *reliable continuous transfer of data* from sensor devices to the server. For example, in a private health care, the patient may be out of coverage area from the smart-phone gateway because of patients coming in and out frequently. This scenario would

Table 1: Technical comparison of different messaging approaches and algorithms being used in several papers of Sensor-Cloud.

| Papers | Using traditional SOAP messages | Using SPWS messages | Using data caching or hop-by-hop processing | Using QoS-based routing | Using SGIM | Using SPMC | Using event-matching algorithms | Using matrix-matching algorithms | Using orthogonal neural network algorithm | Using load-balancing algorithm |
|---|---|---|---|---|---|---|---|---|---|---|
| [59] | No | No | No | No | Yes | Yes | Yes | No | No | No |
| [5] | No | No | Yes | Yes | No | Yes | Yes | No | No | No |
| [34] | No | Yes | No | No | No | No | No | No | No | No |
| [60] | No | No | Yes | No | No | No | Yes | No | No | No |
| [61] | No | No | No | Yes | No | No | Yes | No | No | No |
| [39, 62] | No | No | No | No | No | No | Yes | Yes | No | No |
| [13, 63] | No | No | Yes | No | No | No | No | No | No | No |
| [64] | No | No | No | No | No | No | No | No | Yes | No |
| [65] | No | No | Yes | Yes | No | No | No | No | No | No |
| [37, 66, 67] | Yes | No | No | No | No | No | No | No | No | No |
| [45] | Yes | No | No | No | No | No | No | No | No | Yes |

SPWS: sensor profile for web services.
SGIM: statistical group index matching.
SPMC: stream monitoring and processing component.

be more prone to connection failure between the server and smart phone (or any other display device, like PDA) and thus this scenario must be considered while designing such system in order to avoid accumulation of errors [15, 69].

*7.2. Storage Issues.* Some engineering issues like storage of data at server side and transferring data from phone to server must have to be considered. To tackle this, timestamps are sent with each data packet to assist in reconstruction of data on the server side. Most of the data processing is done at server end so the system must be designed to avoid the bursty processing due to multiple users connected simultaneously to the system. The system must be designed to accommodate multiple users to connect at the same time [15].

Storage issues can be tackled with the introduction of predictive storage concept proposed by the authors in [70]. This concept of storage keeps it easily fit to the correlated behavior of the physical environment and builds an architecture that focuses the sensor data archival at some remote sensors of Sensor-Cloud infrastructure. It also uses predictive caching at proxies.

*7.3. Authorization Issues.* A web-based user interface is used for doctors, patients, helpers, care-givers, and so forth to inspect and analyze the patients' health-related results remotely. Therefore, the system should offer different *authorization roles* for different types of users and authenticated via this web interface. This will enable the *privacy* to some extent by allowing the care givers to restrict them to the patients that he/she will take care of.

*7.4. Power (Battery) Issues.* While using smart phone as a gateway, *power (battery)* is the main issue that has to be taken care of because the continuous processing and wireless transmission would drain out the mobile battery within few hours or days. Thus, it is important to tackle power issues while connecting mobile phone gateway with the Sensor-Cloud infrastructure [45].

*7.5. Event Processing and Management.* Sensor-Cloud has to cope with very complex event processing and management issues [16, 17], such as the following.

(i) How the events have to be synchronized that may come from different sources in different time because of delays in network?

(ii) How the event processing rules have to be changed without affecting the system?

(iii) How the messages and events of varying types are supported?

(iv) How to support the enormous numbers of events and its conditions in an optimal way?

(v) How can we recognize the context (i.e., spatial, temporal, semantic) to its relevant situation detection?

*7.6. Service Level Agreement (SLA) Violation.* Consumers dependency on cloud providers for their applications' computing needs (i.e. their processing, storage, and analysis of enormous sensor data) on demand may require a specific Quality of Service (QoS) to be maintained. But if cloud providers are unable to provide QoS on user's demand even in the case of processing huge sensor data in critical environmental situations, it would result in SLA violation and cloud provider must be responsible for that. So, we need a reliable dynamic collaboration among cloud providers. But opting for the best combination of cloud providers in dynamic collaboration is a big challenge in terms of cost, time, and discrepancy between providers and QoS [38].

*7.7. Need for Efficient Information Dissemination.* In Sensor-Cloud an efficient information dissemination mechanism is needed that can match the published events or sensor data to appropriate user's applications. But there are some issues like maintaining flexibility in providing a powerful subscription schema, which may capture information about events, guaranteeing the scalability with respect to a number of subscribers and published events or sensor data [38]. Since the data sets and their relevant access services are distributed geographically, the allocation of data storage and dissemination becomes critical challenges.

*7.8. Security and Privacy Support Issues.* There are fewer standards available to ensure the integrity of the data in response to change due to authorized transactions. The consumers need to know whether his/her data at cloud center is well encrypted or who supervises the encryption/decryption keys (i.e., the cloud vendor or customer himself). Private health data may become public due to fallacy or inaccuracy; that is, consumer's privacy may be lost into cloud and sensor data or information uploaded into clouds may not be supervised correctly by user. The US WellPoint disclosed that 130,000 records of its consumers had leaked out and become available publicly over the Internet. So better privacy policies are the demand of the time that can offer the services themselves while maintaining the privacy [37, 38].

*7.9. Real-Time Multimedia Content Processing and Massive Scaling.* Usage of large amount of multimedia data and information in real time and its mining is a big challenge in the integration of heterogeneous and massive data sources with cloud. To classify this real-time multimedia information and contents such that it may trigger the relevant services and assist the user in his current location is also a big challenge to be handled [16].

*7.10. Collective Intelligence Harvesting.* The heterogeneous real-time sensor data feed enhances the decision-making capability by using the appropriate data and decision level fusion mechanisms. But maximization of intelligence developed from the massively collocated information in cloud is still a very big challenge [16].

*7.11. Energy Efficiency Issues.* The basic disadvantages of a WSN and cloud computing are almost the same, and energy efficiency of sensor nodes is lost due to the limited storage and processing capacity of nodes. The authors of [37] have proposed a system for health monitoring using the textile sensors, which work much better and give more accurate results. These textile sensors can be easily sewed and are even washable. Although the proposed system of textile sensors is performing well in the majority of aspects, the battery can last only 24 hours after continuous monitoring and data transmitting regarding user's heartbeat rate, movement, respiratory conditions, and so forth. The gathered accumulated data can then be visualized in charts using some web applications and the results are received at user end through an alert message remotely on user's smartphone. But in order to extend system independency, energy efficiency of such systems (textile sensors and microcontroller based) is a primary issue that has to be handled.

Data caching mechanism [71] can be used to reuse bygone sensor data for applications that are tolerant to time, for example, an application related to variant room temperature. If this bygone sensor data is used to satisfy the various requests for a common sensor data, the energy consumption will be reduced [11]. Still more work is needed to overcome the energy consumption.

To improve the energy efficiency and memory usage in a Sensor-Cloud infrastructure, there should be a middleware which can tackle the adverse situation in case of continuous and long-duration monitoring of data. This can be done through the gateway that is acting as a middleware and collects the huge sensor data from sensor nodes [13]. This middleware should be able to compress the sensor data to avoid the transmission load and then transmits it back to the gateway acting as a middleware on cloud side which in turn decompresses and stores it there. When the transmission overload reduces, the energy consumption of sensor nodes improves automatically due to less processing.

*7.12. Bandwidth Limitation.* Bandwidth limitation is one of the current big challenges that has to be handled in Sensor-Cloud system when the number of sensor devices and their cloud users increases dramatically [71]. However, there are a number of optimal and efficient bandwidth allocation methods proposed, but to manage the bandwidth allocation with such a gigantic infrastructure consisting of huge device assets and cloud users, the task of allocating bandwidth to every devices and users becomes very difficult.

*7.13. Network Access Management.* There are various numbers of networks to deal with in Sensor-Cloud architecture applications. So a proper and efficient access management scheme for these networks is needed because this will optimize the bandwidth usage and improve link performance [72].

*7.14. Pricing Issues.* Access to the services of Sensor-Cloud involves both the sensor-service provider (SSP) and cloud-service provider (CSP). However, both SSPs and CSPs have different customer's management, services management, and modes and methods of payments and pricing. So all this together will lead to a number of issues [10] such as

(i) how to set the price?

(ii) how the payment is to be made?

(iii) how the price is be distributed among different entities?

The growing demand for controlling and monitoring the environment and its applications results in the growth of a large number of devices while the cost of deployment and connecting them to heterogeneous network continues to drop. However, the interfaces, protocols, connections, and so forth increase at exponential rate, thereby making it difficult and expensive for information technology (IT) people to integrate the devices (sensor devices) into the cloud world. To eradicate the complexity and cost associated with integrating the sensors into cloud or any highly distributed system, the authors of [34] proposed emerging and existing standards from both domains, for example, embedded sensor and IT domains within a service-oriented sensor architecture (SOSA).

The authors in [34] extend the service oriented paradigm to a sensor network and use the service oriented process parameter (like profiling sensors for web services), which helps in intelligence integration into the Internet. This solution when extended to Sensor-Cloud would result in high availability and reliability. It was found that the energy consumption of sensor nodes reduced drastically when the data exchange is done among sensors into a heterogeneous network with gateway through the SPWS (sensor profiles for web services) as compared to traditional SOAP messages. It has been observed also that the cost of memory usage in sensor nodes remains constant with SPWS whereas it is increased with SOAP messages.

The entity which is most responsible for the cost of Sensor-Cloud service model is the message exchanged among sensors into a heterogeneous network environment. Using SPWS, power consumption of sensor nodes as well as the memory usages in sensor nodes are reduced drastically. In the Sensor-Cloud infrastructure, the cost of communication among sensors is more than the processing cost. Hence the reduction in power consumption and memory usage leads to less communication cost which also enables an energy efficient model of Sensor-Cloud.

*7.15. Interface Standardization Issues.* Web interfaces currently provide the interface among Sensor-Cloud users (may be smartphone users) and cloud. But web interface may cause overhead because the web interfaces are not specifically designed for smart phones or mobile devices. Also, there would be compatibility issues for web interface among devices and in this case signaling, standard protocol, and interface for interacting between Sensor-Cloud users and the cloud would require seamless services for implementation. Thus, interoperability would be a big issue when the Sensor-Cloud users need to access the services with cloud [10].

*7.16. Maintenance Issues.* In order to keep the end users' loyalty, the cloud should cope with the service failure [73]. For this a regular maintenance is needed and redundancy techniques should be implemented to ensure the smooth and continuous flow of services. This can be done by backing up the data regularly and by distributing their multiple data centers geographically across the world.

*7.17. Resource and Hardware Compatibility Issues.* Hardware compatibility as well as software compatibility both can be solved in cloud computing environment by allowing the sharing of hardware or software resources or services [5], but there may be the case when sensor or some other resources being used are lost due to some calamity or severe weather condition. To handle these issues, the authors in [70] have proposed PRESTO architecture that enables users/clients to view the data in a single logical view which is distributed across several sensor proxies and remote sensors. This enables the users to view variability at several lossy levels and unreliable remote sensor network resources. The PRESTO also supports archival queries on data resources that enable the historical data query to prevent any unusual events to better understand them for future application scenarios.

## 8. Conclusion

In this paper, we surveyed the use of Sensor-Cloud architecture in the context of several applications. The Sensor-Cloud architecture enables the sensor data to be categorized, stored, and processed in such a way that it becomes cost-effective, timely available, and easily accessible. Earlier, most WSN systems which were included to several controlling/monitoring schemes were closed in nature, zero, or less interoperability, specific application oriented, and nonextensible. However, integrating the existing sensors with cloud will enable an open, extensible, scalable, interoperable, and easy to use, reconstructible network of sensors for numerous applications. In this paper, we have discussed the opportunities of implementing the technology to handle more complex situations of a real world through the service innovation capability of Sensor-Cloud infrastructure.

## Acknowledgment

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through the research group project no. RGP-VPP-049.

## References

[1] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54–61, 2004.

[2] T. Haenselmann, "Sensor networks," GFDL Wireless Sensor Network textbook 2006.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[4] G. Simon, G. Balogh, G. Pap et al., "Sensor network-based countersniper system," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 1–12, Baltimore, Md, USA, November 2004.

[5] S. K. Dash, J. P. Sahoo, S. Mohapatra, and S. P. Pati, "Sensor-cloud: assimilation of wireless sensor network and the cloud," in *Advances in Computer Science and Information Technology. Networks and Communications*, vol. 84, pp. 455–464, Springer-Link, 2012.

[6] M. Castillo-Effen, D. H. Quintela, R. Jordan, W. Westhoff, and W. Moreno, "Wireless sensor networks for flash-flood alerting," in *Proceedings of the 5th IEEE International Caracas Conference on Devices, Circuits and Systems (ICCDCS '04)*, pp. 142–146, November 2004.

[7] G. Werner-Allen, K. Lorincz, M. Welsh et al., "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing*, vol. 10, no. 2, pp. 18–25, 2006.

[8] M. Yuriyama, T. Kushida, and M. Itakura, "A new model of accelerating service innovation with sensor-cloud infrastructure," in *Proceedings of the annual SRII Global Conference (SRII '11)*, pp. 308–314, 2011.

[9] J. Yick, B. Mukherjee, and D. Ghosal, *Wireless Sensor Network Survey*, Elsevier, 2008.

[10] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, *A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches*, Wireless Communications and Mobile Computing-Wiley Online Library, 2011.

[11] W. Kim, "Cloud computing: today and tomorrow," *Journal of Object Technology*, vol. 8, pp. 65–72, 2009.

[12] M. Yuriyama and T. Kushida, "Sensor-cloud infrastructure physical sensor management with virtualized sensors on cloud computing," in *Proceedings of the IEEE 13th International Conference on Network-Based Information Systems (NBiS '10)*, pp. 1–8, September 2010.

[13] L. P. D. Kumar, S. S. Grace, A. Krishnan, V. M. Manikandan, R. Chinraj, and M. R. Sumalatha, "Data filtering in wireless sensor networks using neural networks for storage in cloud," in *Proceedings of the IEEE International Conference on Recent Trends in Information Technology (ICRTIT '11)*, 2012.

[14] M. O'Brien, *Remote Telemonitoring—A Preliminary Review of Current Evidence*, European Center for Connected Health, 2008.

[15] B. Jit, J. Maniyeri, K. Gopalakrishnan et al., "Processing of wearable sensor data on the cloud—a step towards scaling of continuous monitoring of health and well-being," in *Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC '10)*, pp. 3860–3863, 2010.

[16] http://www.ntu.edu.sg/intellisys.

[17] K. T. Lan, "What's Next? Sensor+Cloud?" in *Proceeding of the 7th International Workshop on Data Management for Sensor Networks*, pp. 978–971, ACM Digital Library, 2010.

[18] Sensor-Cloud, http://sensorcloud.com/system-overview.

[19] C. O. Rolim, F. L. Koch, C. B. Westphall, J. Werner, A. Fracalossi, and G. S. Salvador, "A cloud computing solution for patient's data collection in health care institutions," in *Proceedings of the 2nd International Conference on eHealth, Telemedicine, and Social Medicine (eTELEMED '10)*, pp. 95–99, February 2010.

[20] U. Varshney, "Pervasive healthcare and wireless health monitoring," *Mobile Networks and Applications*, vol. 12, no. 2-3, pp. 113–127, 2007.

[21] LifeShirt, http://www.vivometrics.com/site/system.html.

[22] Smart Shirt, http://www.gtwm.gatech.edu.

[23] U. Varshney, "Managing wireless health monitoring for people with disabilities," *IEEE IT-Professional*, pp. 12–16, 2006.

[24] SensorML, http://www.opengeospatial.org/standards/sensorml.

[25] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, pp. 1–14, November 2009.

[26] A. Rowe, V. Gupta, and R. Rajkumar, "Low-power clock synchronization using electromagnetic energy radiating from AC power lines," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, pp. 211–224, November 2009.

[27] S. Madden and M. J. Franklin, "Fjording the stream: an architecture for queries over streaming sensor data," in *Proceedings of the 18th International Conference on Data Engineering*, pp. 555–566, March 2002.

[28] R. Katsuma, Y. Murata, N. Shibata, K. Yasumoto, and M. Ito, "Extending k-coverage lifetime of wireless sensor networks using mobile sensor nodes," in *Proceedings of the 5th IEEE International Conference on Wireless and Mobile Computing Networking and Communication (WiMob '09)*, pp. 48–54, October 2009.

[29] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *Proceedings of the 9th Internatinal Conference Architectural Support for Programming Languages and Operating Systems*, pp. 93–104, November 2000.

[30] K. Matsumoto, R. Katsuma, N. Shibata, K. Yasumoto, and M. Ito, "Extended abstract: minimizing localization cost with mobile anchor in underwater sensor networks," in *Proceedings of the 4th ACM International Workshop on UnderWater Networks (WUWNet '09)*, November 2009.

[31] T. Sookoor, T. Hnat, P. Hooimeijer, W. Weimer, and K. Whitehouse, "Macrodebugging: global views of distributed program execution," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, pp. 141–154, November 2009.

[32] J. Shneidman, P. Pietzuch, J. Ledlie, M. Roussopoulos, and M. Seltzerand Welsh, "Hourglass: an infrastructure for connecting sensor networks and applications," Harvard Technical Report TR-21-04, 2004.

[33] M. Gaynor, S. L. Moulton, M. Welsh, E. LaCombe, A. Rowan, and J. Wynne, "Integrating wireless sensor networks with the grid," *IEEE Internet Computing*, vol. 8, no. 4, pp. 32–39, 2004.

[34] R. S. Ponmagal and J. Raja, "An extensible cloud architecture model for heterogeneous sensor services," *International Journal of Computer Science and Information Security*, vol. 9, no. 1, 2011.

[35] S. Guo, Z. Zhong, and T. He, "FIND: faulty node detection for wireless sensor networks," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, pp. 253–266, November 2009.

[36] A. Alexe and R. Exhilarasie, "Cloud computing based vehicle tracking information systems," *International Journal of Computer Science and Telecommunications*, vol. 2, no. 1, 2011.

[37] C. Doukas and I. Maglogiannis, "Managing wearable sensor data through cloud computing," in *Proceedings of the IEEE 3rd International Conference on Cloud Computing*, 2011.

[38] X. H. Le, "Secured WSN-integrated cloud computing for u-life care," in *Proceedings of the Consumer Communications and networking Conference (CCNC '10)*, pp. 1–2, IEEE, 2010.

[39] T.-D. Nguyen and E.-N. Huh, "An efficient key management for secure multicast in Sensor-Cloud"," in *Proceedings of the IEEE 1st ACIS/JNU International Conference on Computers, Networks, Systems, and Industrial Engineering*, 2011.

[40] Nimbits Data Logging Cloud Sever, http://www.nimbits.com.

[41] Pachube Feed Cloud Service, http://www.pachube.com.

[42] iDigi—Device Cloud, http://www.idigi.com.

[43] IoT—ThingSpeak, http://www.thingspeak.com.

[44] G. Demiris, B. K. Hensel, M. Skubic, and M. Rantz, "Senior residents' perceived need of and preferences for "smart home" sensor technologies," *International Journal of Technology Assessment in Health Care*, vol. 24, no. 1, pp. 120–124, 2008.

[45] K. Lee, D. Murray, D. Hughes, and W. Joosen, "Extending sensor networks into the Cloud using Amazon web services," in *Proceedings of the 1st IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA '10)*, pp. 1–7, November 2010.

[46] B. Jit, J. Maniyeri, S. Louis, K. Gopalakrishnan, and P. Yap, "Design and trial deployment of a practical sleep activity pattern monitoring system," in *Proceedings of the International Conference on Smart Homes and Health Telematics (ICOST '09)*, Tours, France, June 2009.

[47] B. Jit, J. Maniyeri, S. Louis, and L. K. P. Yap, "Fast matching of sensor data with manual observations," in *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society: Engineering the Future of Biomedicine (EMBC '09)*, pp. 1675–1678, September 2009.

[48] N. Kurata, M. Suzuki, S. Saruwatari, and H. Morikawa, "Actual application of ubiquitous structural monitoring system using wireless sensor networks," in *Proceedings of the 14th World Conference on Earthquake Engineering (WCEE '08)*, 2008.

[49] Google Health, http://www.google.com/health.

[50] Korea u-Life care system.

[51] http://www.apan.net/meetings/HongKong2011/Session/Agriculture.php/.

[52] H. H. Tran and K. J. Wong, "Mesh networking for seismic monitoring—the sumatran cGPS array case study," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '09)*, April 2009.

[53] Tunnel Monitoring System: http://www.advantech.com/intelligent-automation/Industry%20Focus/%7BC274D52C-95D2-499E-9E16-6C1F41D1CD6/.

[54] N. B. Priyantha, A. Kansal, M. Goraczko, and F. Zhao, "Tiny web services: design and implementation of interoperable and evolvable sensor networks," in *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys '08)*, pp. 253–266, Raleigh, NC, USA, 2008.

[55] N. Hoeller, C. Reinke, J. Neumann, S. Groppe, D. Boeckmann, and V. Linnemann, "Efficient XML usage within wireless sensor networks," in *Proceedings of the 4th Annual International Conference on Wireless Internet (WICON '08)*, pp. 17–19, November 2008.

[56] A. Triantafyllidis, V. Koutkias, I. Chouvarda, and N. Maglaveras, "An open and reconfigurable Wireless Sensor Network for pervasive health monitoring," in *Proceedings of the 2nd International Conference on Pervasive Computing Technologies for Healthcare*, pp. 112–115, February 2008.

[57] K. Lee and D. Hughes, "System architecture directions for tangible cloud computing," in *Proceedings of the International Workshop on Information Security and Applications (IWISA '10)*, Qinhuangdao, China, October 2010.

[58] I. Gul and M. Hussain, "Distributed cloud intrusion detection model," *International Journal of Advanced Science and Engineering Technology*, vol. 34, 2011.

[59] M. M. Hassan, B. Song, and E. N. Huh, "A framework of sensor—cloud integration opportunities and challenges," in *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication (ICUIMC '09)*, pp. 618–626, ACM, Suwon, Republic of Korea, January 2009.

[60] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, "Building efficient wireless sensornetworks with low-level naming," in *Proceedings of the Symposium on Operating Systems Principles*, pp. 146–159, ACM, Alberta, Canada, October 2001.

[61] S. K. Dash, S. Mohapatra, and P. K. Pattnaik, "A survey on applications of Wireless Sensor Network using Cloud Computing," *International Journal of Computer Science and Emerging Technologies*, vol. 1, no. 4, 2010.

[62] K. Ahmed and M. Gregory, "Integrating wireless sensor networks with cloud computing," in *Proceedings of the 7th International Conference on Mobile Ad-hoc and Sensor Networks (MSN '11)*, pp. 364–366, Beijing, China, 2011.

[63] R. Ian, C. Michele, J. Ho Young, M. Zoltan, and A. Karl, "Building a front end interface for sensor data cloud," in *Proceedings of the 2nd International Conference on Signals, Systems and Automation (ICSSA '11)*, vol. 6784 of *Lecture Note in Computer Science*, 2011.

[64] J. Cen, T. Yu, Z. Li, S. Jin, and S. Liu, "Developing a disaster surveillance system based on wireless sensor network and cloud platform," in *Proceedings of the IET International Conference on Communication Technology and Application (ICCTA '11)*, pp. 757–761, Beijing, China, 2012.

[65] M. Baktashmotlagh, A. Bigdeli, and B. C. Lovell, "Dynamic resource aware sensor networks: Integration of sensor cloud and ERPs," in *Proceedings of the 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS '11)*, pp. 455–460, Klagenfurt, Austria, 2011.

[66] V. V. Tan, D. S. Yoo, and M. J. Yi, "Design and implementation of Web service by using OPC XML-DA and OPC complex data for automation and control systems," in *Proceedings of the 6th IEEE International Conference on Computer and Information Technology (CIT '06)*, p. 263, IEEE, September 2006.

[67] V. Rajesh, J. M. Gnanasekar, R. S. Ponmagal, and P. Anbalagan, "Integration of wireless sensor network with cloud," in *Proceedings of the International Conference on Recent Trends in Information, Telecommunication, and Computing (ITC '10)*, pp. 321–323, March 2010.

[68] C. O. Rolim, F. L. Koch, A. Sekkaki, and C. B. Westphall, "Telemedicine with grids and wireless sensors networks," in *Proceedings of the International Conference on e-Medical Systems (e-Medisys '08)*, 2008.

[69] G. Singh, J. O'Donoghue, and C. K. Soon, "Telemedicine: issues and Implications," *Technology and Health Care*, vol. 10, no. 1, pp. 1–10, 2002.

[70] M. M. Islam, M. M. Hassan, G. W. Lee, and E. N. Huh, "A survey on virtualization of wireless sensor networks," *Sensors Journal*, vol. 12, no. 2, pp. 2175–2207, 2012.

[71] Y. Xu, S. Helal, T. My Thai, and M. Schmalz, "Optimizing push/pull envelopes for energy-efficient cloud-sensor systems," in *Proceedings of the 14th ACM international conference (MSWiM '11)*, 2011.

[72] F. Ge, H. Lin, A. Khajeh et al., "Cognitive radio rides on the cloud," in *Proceedings of the IEEE Military Communications Conference (MILCOM '10)*, pp. 1448–1453, November 2010.

[73] R. Liu and I. J. Wassell, "Opportunities and challenges of wireless sensor networks using cloud services," in *Proceedings of the Workshop on Internet of Things and Service Platforms (IoTSP '11)*, Tokyo, Japan, December 2011.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 958061, 14 pages
http://dx.doi.org/10.1155/2013/958061

*Research Article*

# Architecture and Routing Protocols for Smart Wireless Home Sensor Networks

**Yang Xu,[1] Shuai Wu,[1] Ruochen Tan,[1] Zheng Chen,[1] Min Zha,[2] and Tina Tsou[2]**

[1] *University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*
[2] *Huawei Technologies Co., Ltd., Shenzhen, Guangdong 518129, China*

Correspondence should be addressed to Yang Xu; xuyang.uestc@gmail.com

Received 25 May 2012; Revised 29 September 2012; Accepted 18 October 2012

Academic Editor: Regina B. Araujo



As an important application domain of wireless sensor networks (WSN), wireless home sensor network (WHSN) can be built as a traditional WSN. However, when we consider its own character that plug-in sensors are fixed with AC power supply while mobile sensors are battery powered, traditional WSN techniques do not match well. In this paper, we propose a smart wireless home sensor network architecture with improved routing protocols. It is a hierarchical architecture in which AC-powered sensors act as backbone nodes for data retransmission, while battery-powered sensors act as leaves that only transmit data relevant to themselves. Each sensor is assigned with a prime number as its location identifier. For our routing algorithm, the LID is used as routing address and is decomposed to a sequence of prime numbers that indicates the route towards its destination. In addition, we improve existing routing algorithms such as SPIN, LEACH, and DD to incorporate traditional WSN routing algorithms into our smart WHSN architecture, and comparable efficiencies are made between them. Moreover, we propose a network path recovery algorithm for failures that are caused by node mobility or backbone node failures. All our algorithms have been provided with faithful simulations to verify the feasibility and efficiency.

## 1. Introduction

Wireless home sensor networks (WHSNs) that connect household appliances to the Internet become important application domains of wireless sensor network (WSN) [1, 2]. The objective is to use kinds of wireless techniques to interconnect household electrical devices and the sensors to form a unified platform so that all smart household devices can communicate and be controlled cooperatively, securely, and robustly. There have been extensive studies made in this field to build an autoadaptive and green wireless home sensor network. Azim and Islam [3] proposed a hybrid LEACH [3, 4] algorithm to build a relay node in its hierarchical clusters so that an energy-saving WHSN can be realized. Park and Corson [5] proposed a highly adaptive distributed routing algorithm for WHSN by using the SPIN [5] protocol. Zhang et al. [6] proposed an energy efficient approach called directed diffusion (DD) [6] protocol based on passive clustering. Although these solutions have been proven to be more efficient than traditional WSN protocols when applied in home sensor networks, they only took partial advantages of the WHSN features and made unnecessary assumptions such as full mobility of WHSN nodes. Therefore, based on the state of the art analysis, a comprehensive approach that matches the needs of smart wireless home sensor network is still on the way.

In this paper, we investigated the key features of WHSN that distinguish it from traditional WSNs and proposed a novel algorithm to make a full use of those features. First of all, some nodes of WHSN are embedded in household appliances and connect AC as power supply, which are not similar with typical battery-powered WSN nodes. Hence, energy consumption is no longer a constraint for AC-powered nodes, and they might be able to make more contribution on packet transmissions than those energy-constrained mobile nodes. Secondly, the deployment of WHSN is partially fixed other than traditional WSNs where most nodes are randomly scattered. For example, temperature sensors in WHSN are adhered on the wall or embedded into the household appliances that are permanently immovable. Therefore, we can

model the WHSN architecture as a semistructured layout rather than a typical mobile ad hoc network. Thirdly, sensor nodes in WHSNs [7, 8] typically range from 100 to 1000 in a household, which is categorized as a middle size WSN. Therefore, scalability is not a major bottleneck in network design for WHSNs.

By making a good use of these key features to build a flexible and efficient wireless home sensor network, we proposed a hierarchical architecture for WHSN. In this architecture, sensor nodes are categorized into two classes: plug-in nodes that are energy restriction free but localization fixed and energy-restricted nodes that are modeled with mobilities. According to our design, nodes are organized as a tree structure where the gateway acts as the root. To take the advantage of plug-in nodes, we deployed them as backbones of the tree for data retransmissions, and their immovability is helpful to keep the tree structure stable. As energy-restricted nodes are normally movable, such as sensors in a robot, they are more suitable to be connected as leaf nodes in the network.

Based on the architecture design, we are able to customize simple and flexible routing protocols for WHSN. The key innovation is that we excluded the traditional routing table, by assigning each node with a prime-numbered address, and each routing path is identified with a location identifier (LID) which is the production of the prime addresses of all the relay nodes. The key advantage of this design is that when a node chooses the next relay node, it can easily adopt a decomposing operation that only the next transmitter can be divided. When the package is transmitted, the LID is replaced with the quotient to identify the remaining path, and when the LID is a prime number, it denotes that it is the address of the destination node. Therefore, the path can be easily discovered by the relay nodes without routing table.

In addition to our new algorithm design, we incorporated traditional WSN protocols into our smart home sensor networks architecture by improving some popular routing algorithms such as SPIN, LEACH, and DD [6, 9]. The key is to take the advantage of both the hierarchical architecture and energy restriction free plug-in nodes. In the end, we introduced our network maintenance algorithm. Benefited by the WHSN features, our protocols can improve the routing efficiency and effectively reduce the cost for network reorganization caused by the movement of mobile nodes or sensor node failures.

## 2. Smart WHSN Architecture

In this section, we introduce our semistructured WHSN architecture. Our design is based on the key features on where and how sensors are deployed in a household. The application domain can be specifically illustrated as follows:

(1) there is a home gateway acting as the sink node to the internet;

(2) some nodes are embedded in household appliances and directly connected with AC adapter. Due to its cable connection, it is hardly movable;

(3) the other nodes are most likely moveable, such as sensors in a robot. Due to their mobilities, they are battery powered, and energy consumption is a bottleneck for data transmission;

(4) wireless home sensor networks are typically with hundreds of sensors.

*2.1. Plug-In Nodes and Mobile Nodes.* In a household, plug-in nodes are normally location fixed and integrated into household appliances with AC supply. For example, in Figure 1, the refrigerator contains temperature sensors and radio frequency (RF) sensors and they are all AC powered, which means they have no energy consumption limitations and can keep awake all the time. In WHSN, energy consumption is key factor to affect the lifetime of the network. If we can make use of plug-in nodes as backbone for data transmission, we can significantly reduce energy consumption of mobile nodes and extend the WHSN lifetime. In addition, plug-in nodes with power supply are normally more stable than mobile nodes. If they are used as backbone nodes, it will help improve the stability of WHSN from power-off failures.

Other than plug-in nodes, there are plenty of sensors that are movable and powered with battery. Typical mobile sensors include video cameras in robots and localization sensors in vacuum cleaners. The mobility feature of mobile sensors will make necessary changes on network topology, and as they are always battery powered, energy constraint is a key bottleneck for them involving data retransmissions. For example, as shown in Figure 2, modules of the mobile node such as the data acquisition, control, movement, and data transmission are all powered with battery.

*2.2. WHSN Network Design.* The key of our WHSN architecture design is to make the advantage of features that discriminates WHSNs from traditional WSNs. As explained previously, plug-in nodes are more suitable as backbone nodes to take more responsibility of data transmission according to their robustness and battery free features. Moreover, the immovability feature of the plug-in nodes help to keep the backbone stable. On the other hand, mobile nodes powered by battery should connect with the network loosely so that the network will not be significantly changed when they swift their positions, and most importantly, less data retransmissions via mobile nodes will extend their battery lifetime.

According to WHSN features, we proposed a hierarchical architecture for wireless home sensor networks as shown in Figure 3. In our design, nodes are organized as a tree, and the gateway is set as the root of the tree. In this tree, upper layers are most likely to be composed of fixed plug-in sensor nodes, while battery-powered nodes or mobile nodes are more likely to be set as the leaf nodes. Staring with an existing root, the network construction process can be briefly described in Algorithm 1. In this algorithm, nodes are recursively joined to the tree. Firstly, the pending node evaluates its own capability according to its energy constraint, bandwidth, and so forth (line 2). Next, it enumerates its neighbors, queries the capabilities of the neighbors, and reorders its neighbors by their capabilities (lines 3 and 4). With the ordered neighbor list, a node which is of the least capability than the pending

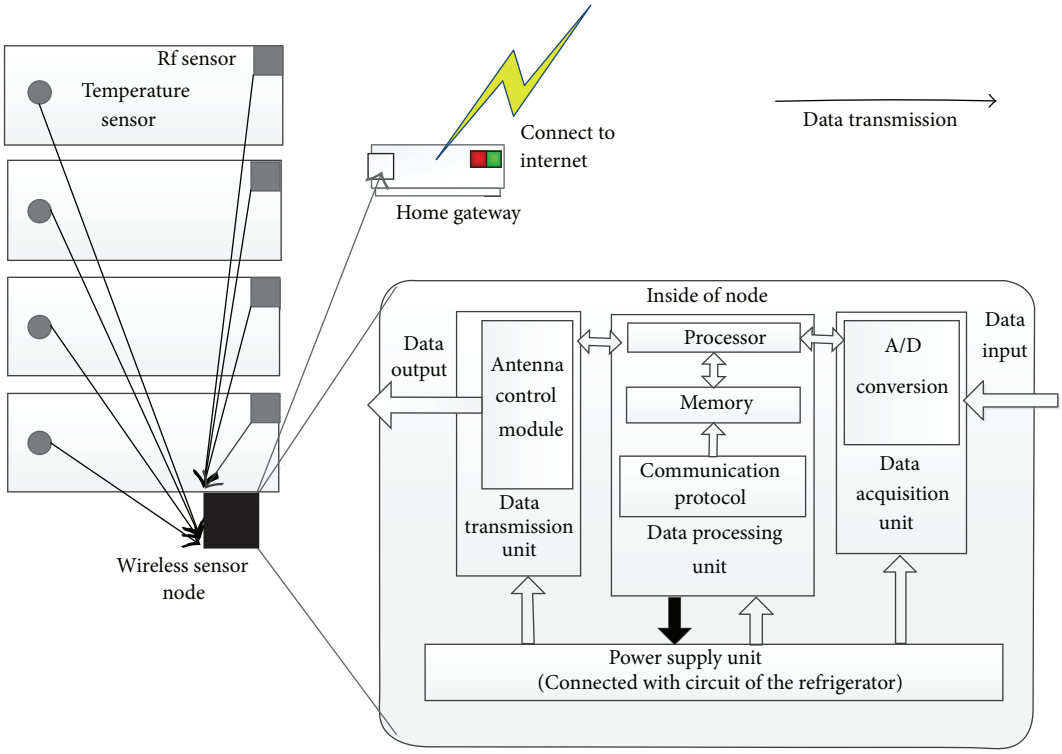



Figure 1: Smart refrigerator, an example of AC-powered plug-in nodes.

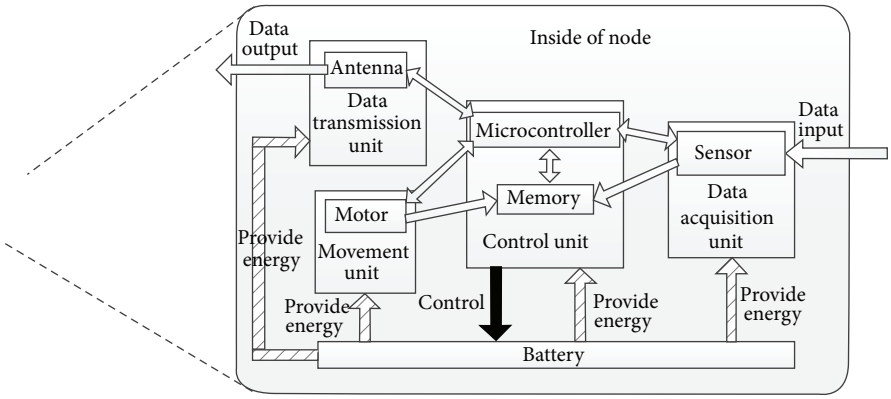



Figure 2: Smart cleaner, an example of battery-powered mobile nodes.

one and does not reach its maximum children number can be found (lines 5–9). As the last step, the pending node will be allocated with a prime-numbered address (line 10) and its LID (line 11). The addressing process will be described in the next section.

Following our network architecture design, we will describe our system design. Although WHSN is a typical WSN where popular WSN protocols can be applied straightforward, we can make necessary revisions to enhance its efficiency. As shown in Figure 4, there are four layers. Based on our hierarchical architecture in the lowest layer, we proposed a prime-numbered addressing protocol and setup for all sensors to form a logical WHSN. In routing protocol layer, we can either modify the traditional routing protocols such as SPIN, LEACH, and DD to match the feature of WHSN or build our new routing protocol as D-HiPr. As the top layer, network maintenance is key protocol to keep the mobile feature of WHSN.

## 3. Addressing and Routing Algorithms

When sensors in WHSN can be organized as a hierarchical architecture as we have proposed, the key is how we can design the addressing and routing protocols that take the merits of tree-like organization as the features of plug-in and mobile nodes. In this section, we present our novel algorithm of dynamical hierarchical routing protocol with prime numbers (D-HiPr) addressing.

```
(1)  for each pending node_i do
(2)      Cap ← node_i · getCap();
(3)      neighbors ← node_i · getNeighbors();
(4)      neighbors ← sortbyCap(neighbors);
(5)      parent ← getCloseCapNb(neighbors, Cap);
(6)      while getChildNum(parent) ⩾
         MaxAllowed do
(7)          neighbors · delete(parent);
(8)          parent ← getCloseCapNb(neighbors);
(9)      end while
(10)     node_i · parent ← parent;
(11)     node_i · address ← assignAddress();
(12)     node_i · LID ← assignLID(parent · LID);
(13) end for
```

Algorithm 1: Network construction.

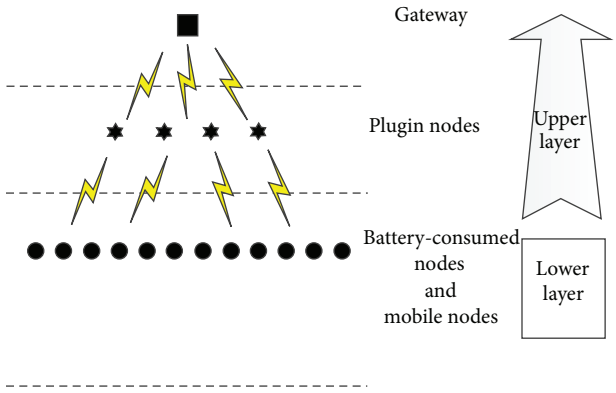



Figure 3: Hierarchical network design for WHSN.

*3.1. Prime-Numbered Addressing Protocol.* In our approach, each node in WHSN is assigned with a prime number as its short address, and any path can be identified by the location identifier (LID), which is the production of the prime addresses of all nodes in the path. The advantage of D-HiPr with prime number addressing lies on the following definition.

**Lemma 1**. *The product of all prime numbers in set P can only be divided by the product of prime numbers which are the subset of P.*

Therefore, each LID can be decomposed to a sequence of prime numbers that identifies a unique path in the WHSN tree from the root, if all the intermediate nodes in this path have different prime-numbered addresses. When a node checks who is the next relayer, it can simply adopt a decomposing operation with its children that only the address of the next transmitter could be divided. In our D-HiPr protocol, two parameters are required for a given node: a prime numbered 16-bit address $A_p$ and LID $Q$ which is the product of nodes in the path linking it from the root. For example, if the path from a node $c$ to the root is $A_1, A_2, \ldots, A_c$, the LID of this node is $Q_c = A_1 \cdot A_2 \cdot \cdots \cdot A_c$. The location of a node can be identified by its parent node, and the node can get its path from its own LID. If the LID of its parent node is $Q_p$, the LID of node $c$ is calculated as $Q_c = A_c \cdot Q_p$. For example, as shown in Figure 5, LID $1955 = 1 \times 5 \times 17 \times 23$ explicitly identifies the path from the root to node 23. When node 5 gains the LID, it will only find node 17 as its next relayer since only 17 can divide the LID.

In order to avoid address conflicts, the gateway responses of all the address assignment. The allocation process is briefly shown as Figure 6. There are three major phases. The first is the neighbor discovery (ND) process [10], which includes router solicitation (RS) and router advertisement (RA). The second phase is for node registrations including two new ICMPv6 messages: node registration (NR) and node confirmation (NC) [11]. Unlike the traditional multi-hop registration, NC includes a location option to contains the LID of its parent node so that its children can calculate their own LID by producing its prime-numbered address with their parent's LID. The third phase is for location registration. The node will use location registration (LR) and location confirmation (LC) messages to inform its LID to the parent node as well as the gateway. Each node in our design uses "hello" message to check whether the neighbors keep the positions. When a node is moved, it will ask the gateway to rejoin the network with a new address. If a node detects that its parent node is gone or a child is moved, network maintenance will start and we will describe that in Section 5.

*3.2. Routing Protocol of D-HiPr.* In this section, we represent the details of D-HiPr and how the prime-numbered address works. For example, when a packet is required to be sent from node C to the destination node D, the D-HiPr routing protocol is explicitly described as Algorithm 2. In this case, the parent node of C is P and its children are $k_i$ ($i = 1, 2, 3, \ldots, n$) where $n$ is the number of its children. The LIDs of node C and node D are $Q_c$ and $Q_d$, and those of node P and their children node $k_i$ are $Q_p$ and $Q_{k_i}$, respectively. To route the packet in node C, the difference between the LID of the destination and its source node is obtained first

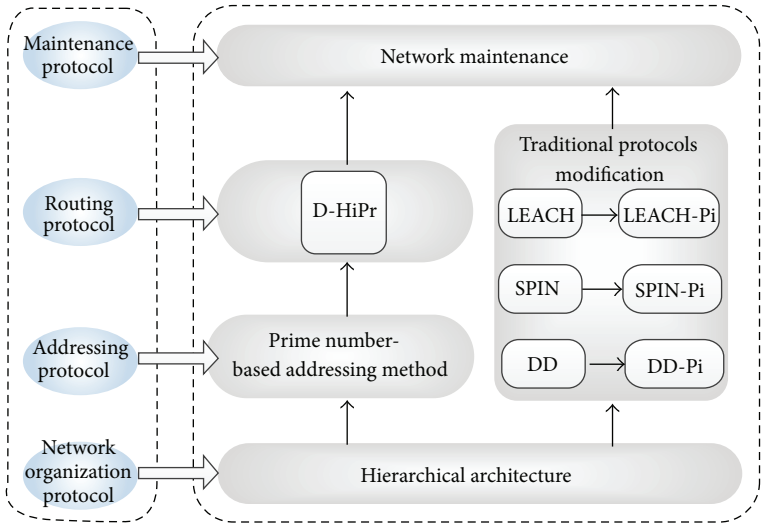



Figure 4: System architecture of a smart WHSN.

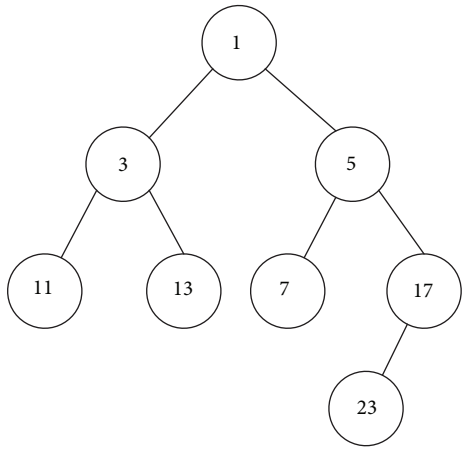



Figure 5: An example of prime-numbered addressing.

(line 2). If the LID of the destination is larger than that of the source, the packet is passed down to the children of node C (line 3). According to the construction of the tree and the creation of LID, the node of which the LID can divide that of the destination is on the path leading the packet directly to its destination. Thus, the right child for the next hop can be chosen (lines 4–7). If there is no child that can pass the packet to its destination, the packet is discarded (line 9). When node C is exactly the destination of the packet, it just holds this packet. If the LID of the destination is smaller than that of the source, the packet is passed to the ancestors of node C and it is directly sent to the parent of C (line 15).

D-HiPr is a novel routing technique by using prime numbers as the sensor nodes' 16-bit local address. Its key advantages meet the requirements of the smart wireless home sensor networks well.

(i) *Nontable-Driven*. By taking the advantage of prime number, D-HiPr uses a single LID for packet routing other than traditional routing table. This mechanism helps to save memory of the transmission nodes as well as reduce traffic delays.

(ii) *Energy Saving*. D-HiPr uses a very simple way to compute routing path and packets in smart WHSN can arrive at the receiver faster and more computation saving to save energy of the relay nodes even it is battery powered.

(iii) *Mobility*. D-HiPr needs no routing table, and the nature of the prime numbers makes the address allocation more flexible. Therefore, D-HiPr can support the mobility of nodes better than traditional routing table.

(iv) *End-to-end Communication*. In smart WHSN with D-HiPr, not all the packets need to be sent via the gateway. In real domain, most packets are directly passed between sensor nodes, and it improves the efficiency of smart WHSN to reduce the traffic of the gateway.

(v) *Medium Scalability*. Although the scalability of prime-numbered addressing is normally hundreds, however, it matches the scalability of WHSN well as it is always composed of hundreds of sensors as well.

## 4. Incorporating WSN Protocols into WHSN

Based on our hierarchical architecture, in the routing protocol layer, we can use D-HiPr or modify existing WSN routing algorithms in WHSN. Our architecture is compatible with different routing protocols. In this section, we modify three typical WSN routing protocols, LEACH, SPIN, and DD. The key of our improvement is to take the advantage of WHSN hierarchical design by mainly using the fixed plug-in nodes instead of mobile nodes to the response for data transportation so that energy consumption bottleneck of mobile nodes can be avoided and the lifetime of the network can be improved.

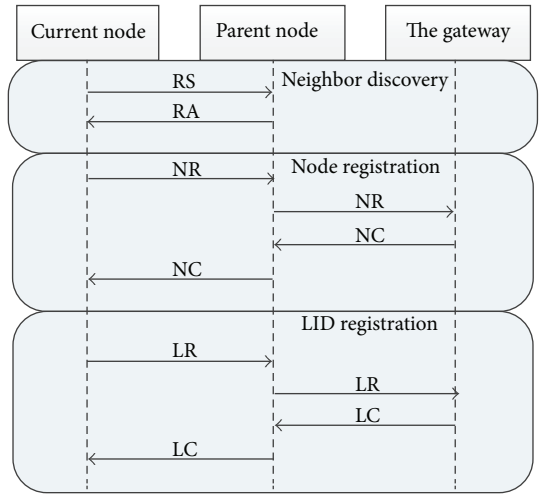



Figure 6: Address allocation process.

```
(1)  for each Node C which holds the packet do
(2)      Comp = packet · Q_d − Q_c;
(3)      if Comp > 0 then
(4)          for i from 1 to n do
(5)              j = Mod(packet · Q_d, Q_{k_i});
(6)              if j = 0 then
(7)                  send(k_i, packet);
(8)              else
(9)                  discard(packet);
(10)             end if
(11)         end for
(12)     else ifComp = 0 then
(13)         hold(packet);
(14)     else
(15)         send(P, packet);
(16)     end if
(17) end for
```

Algorithm 2: Routing process of D-HiPr.

*4.1. LEACH-Pi.* Low energy adaptive clustering hierarchy (LEACH) is a low power consumption adaptive clustering routing protocol designed for wireless sensor network. It is based on monolayer clusters and is a data-centric protocol. The key ideal of LAECH is to choose a cluster head randomly in cycle and evenly distribute the global network load to each sensor node. In this way, network energy consumption and the overall survival time of the network can be improved. LEACH routing protocol is composed of two phases, setup phase and ready phase, and each round includes two phases. To reduce protocol overhead, ready phase should be extended longer than setup phase. In the setup phase, each node randomly generates a random value between (0, 1). If it is less than the threshold $T(n)$ based on the next formula, the node will be selected as a cluster head:

$$T(n) = \begin{cases} \dfrac{p}{1 - p * \left(r \bmod \left(1/p\right)\right)} & \text{if } n \in G, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

As shown in (1), $p$ is the percentage of a node to be selected as the cluster head. $r$ is the round number. $G$ is the group of nodes which are not selected as cluster head in the last $1/p$ round. When the cluster head has been chosen, it broadcasts the message to inform its cluster head position. Nodes decide to join the cluster or not based on the strength of the received message. After a cluster head is chosen, the cluster head will continue to build its cluster. In WHSN, as we explained, node distribution is uneven, and it is unlikely for each node to have an equal chance to be the cluster head. Meanwhile, the WHSN plug-in nodes are more suitable to be a cluster head than randomly chosen node, that is, mobile nodes. It is the key that LEACH protocol should be revised to fit the features of WHSN. By bringing the concept of fixed plug-in nodes in the LEACH protocol, we build the LEACH-Pi where cluster heads are always assigned to plug-in nodes. The process of LEACH-Pi is shown in Algorithm 3.

```
(1)  for each node in WHSN do
(2)     if (node · feature = plug-in) then
(3)        node · state ← head;
(4)        broadcast(Head_Msg);
(5)        Join_Msgs ←receiveJoinMsgs();
(6)        for Join_Msg ∈ Joiln_Msgs do
(7)           node · clusterMb · add(Join_Msg · sender);
(8)        end for
(9)     else
(10)       node · state ← plain;
(11)       Head_Msgs ←receiveHeadMsgs();
(12)       if Head_Msgs ≥ ∅&& state = plain
           then
(13)          sender ← Head_Msgs(0) · sender();
(14)          node · clusterhead ← sender;
(15)          Join_Msg ← "Iam in.";
(16)          send(sender, Join_Msg);
(17)       end if
(18)    end if
(19) end for
```

Algorithm 3: Network building of LEACH-Pi.

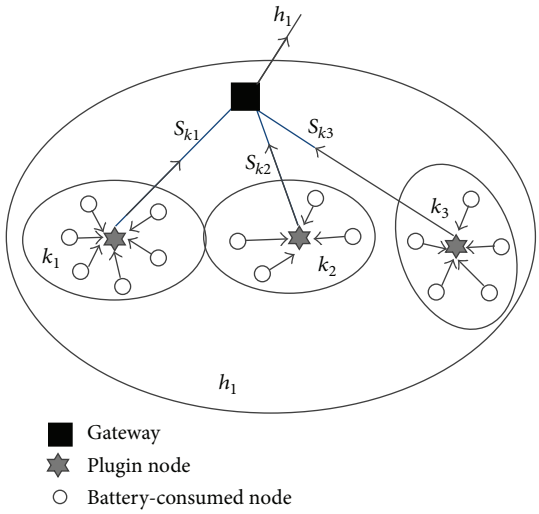



Figure 7: LEACH-Pi.

At first, each plug-in node broadcasts massages to clarify its desire to be cluster header (line 3). Then, it waits for others to join its cluster. After it receives the confirmation messages from its cluster members, the header add them to its cluster member list (lines 6–8). For the nonplug-in nodes, they will receive many header clarifications from cluster headers (line 11). And the first one that sends the clarification will be set to its cluster header (lines 13 and 14). Finally, the non-plug-in node will send a response message ("I am in") to the header (lines 15 and 16). LEACH-Pi routing process is shown in Figure 7.

In the classic LEACH protocol, the most part of energy consumption is to communicate with the cluster heads. In LEACH-Pi, since the plug-in nodes are energy constrained free, their energy consumptions for sending a packet can be neglected. Its energy consumption for communication with cluster heads can be significantly reduced, and the energy consumption of WHSN is merely to communicate between the battery-consumed nodes and cluster heads. Comparing with LEACH, our improved protocol by incorporating the WHSN feature can notably enhance the lifetime of the sensor network bottlenecked by power.

*4.2. SPIN-Pi.* Sensor protocol for information via negotiation (SPIN) is an adaptive data-centric communication protocol. Its goal is to solve the "implosion" and "overlap" in flooding by using the consultation system and the resource adaptive mechanism between nodes. There are three kinds of data packets in SPIN: ADV, REQ, and DATA. Node uses the ADV to announce that it has data to send, use the REQ to request expects to receive data, and use the DATA to package data [12].

In SPIN routing protocol, as Figure 8 indicates, when a SPIN node obtains new data to share, it broadcasts an ADV request message to clarify its demand. If a neighbor is interested in routing this data, it sends an REQ message as a response, and the DATA is sent to this neighbor node. The SPIN protocol does not take the plug-in nodes into consideration and works well with battery-powered nodes. But in WHSN, there are many plug-in nodes which cannot be ignored and SPIN protocol works with low efficiency.

To improve the routing efficiency and extend the network life, we propose the SPIN-Pi routing protocol based on the original SPIN protocol. As shown in Figure 8, the SPIN-Pi protocol mainly takes advantage of the plug-in node and puts most of the routing tasks on them. In SPIN-Pi routing protocol, for the node that has a packet routing request, the routing process is shown in Algorithm 4. When a SPIN node

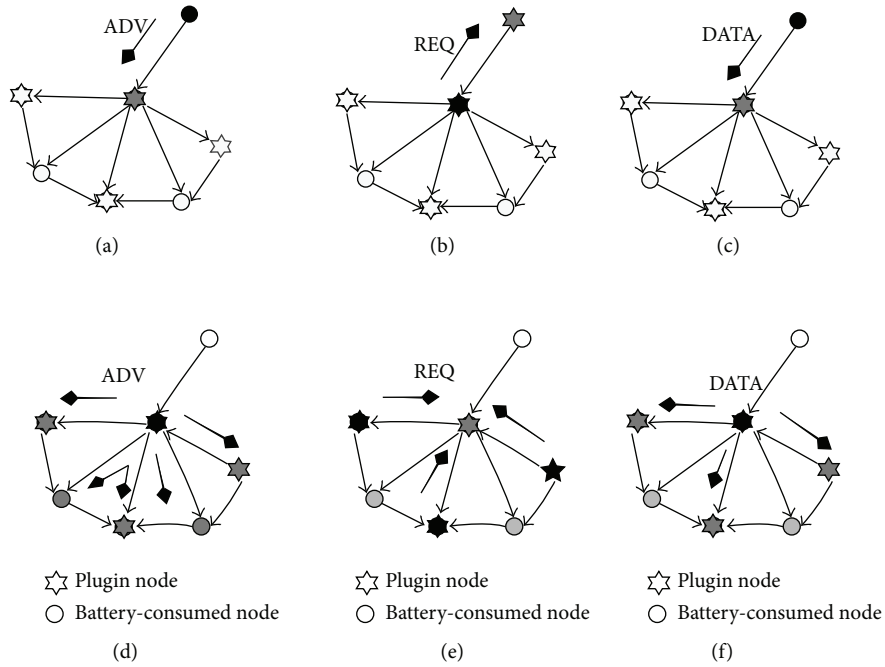



Figure 8: SPIN-Pi.

```
(1)  for each node in WHSN do
(2)     if node · getDATA() ≠ ∅ then
(3)        node · broadcast(ADV);
(4)        REQs ← receiveREQs();
(5)        if REQs ≠ ∅ then
(6)           node · send(REQs · getSend(), DATA);
(7)        else
(8)           node · broadcast(ADV);
(9)           REQs ← receiveREQs();
(10)          if REQs ≠ ∅ then
(11)             node · send(REQs · getSend(), DATA);
(12)          end if
(13)       end if
(14)    end if
(15) end for
```

Algorithm 4: Packet routing request process.

has data to share (line 2), it broadcasts an ADV request message to clarify its desire (line 3). When it receives REQ responses from neighbors (line 4), the packet routing request could be satisfied by these neighbors, and the packet is sent to them (line 6). If there is no one that is willing to route this data, the SPIN node will broadcast an ADV request at the second time (lines 9–11).

For the node that can potentially serve packet routing request, the process in SPIN protocol is shown in Algorithm 5. As the algorithm shows, the plug-in node will serve any ADV request while the non-plug-in node will only route the packet when there is no one willing to serve it in the first request round (lines 5 and 6). Comparing with original SPIN protocol, the energy consumption over the entire network has been significantly reduced by using the plug-in nodes as REQ sender while avoiding mobile nodes involving unnecessary communication.

*4.3. DD-Pi.* Directed diffusion (DD) is a data-centric routing protocol. The routing process is initiated by the information collection node (the gateway of the WHSN). The information collection node broadcasts *interests* packets to the WSN periodically to tell the network what kind of information it wants to collect. And each node which has received *interests* packet will create a *gradient* for this kind of information. Thus, the gradients for all the information in which the gateway is interested are built in this network. When a piece of information is generated in this network, it will be passed to the gateway according to its gradients.

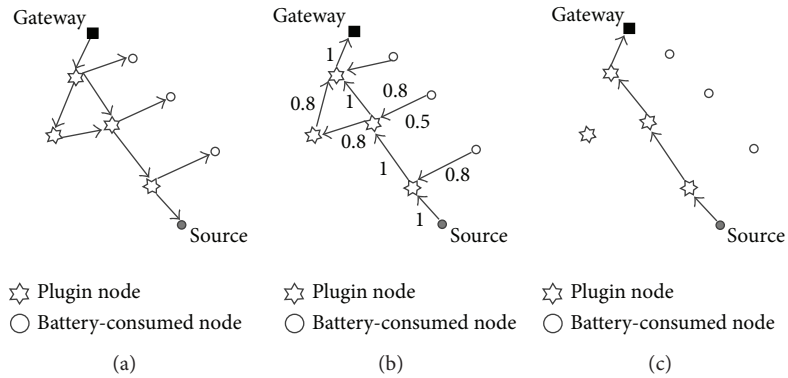



Figure 9: DD-Pi.

```
(1)  for each node in WHSN do
(2)      ADVs ← node · receiveADVs();
(3)      if ADVs ≠ ∅ then
(4)          for ADV ∈ ADVs do
(5)              if (node · feature = plug-in and
                 isFirstRecv(ADV)) or
                 isSecondRecv(ADV) then
(6)                  node · send(ADVs · getSend(), REQ);
(7)              end if
(8)          end for
(9)      end if
(10) end for
```

Algorithm 5: Packet routing response process.

DD protocol is efficient in gathering information for the gateway. To apply this method in WHSN, we propose a modified version of DD protocol called DD-Pi. In this protocol, we define the gradient as the current capability of this node, and the node with a higher capability such as plug-in node has a higher gradient as Figure 9 shows. In this way, most of the communication load can be moved to those plug-in nodes, free of energy consumption bottleneck. Therefore, the network life can be extended.

The DD-Pi routing process is described in Algorithm 6. For each node in WHSN, when an interest message arrives (line 2), the node builds the gradient for this kind of message according to the current capability of the node (line 3). If the current node is a plug-in node (line 4), it will broadcast the interest messages it receives to its neighbors (lines 5 and 6).

## 5. Network Maintenance

Based on our hierarchical architecture, the path connecting a node to the gateway is unique without any alternative route. Therefore, when any node or link is removed or failed, the network is no longer connected and a fast network maintenance schema responding to node mobilities and failures is imperative to build stable and well-performance wireless home sensor network. Although there have been some path recovery mechanisms used in WSN [13, 14], they may be inefficient and costly according to WHSN architecture and routing protocols. In this section, we introduce mobile node maintenance and path recovery algorithm based on our WHSN architecture.

*5.1. Responding to WHSN Mobility.* In our wireless home sensor network design, each node is assigned with a prime-numbered address and a unique LID to denote the path to the root. Moreover, the node with mobility is most likely to be connected as a leaf in the tree. When a leaf is moved, we can very easily deal with its motion according to Algorithm 1 that the moved node is deemed as a newly joined node. The only difference is that it has its prime-numbered address other than being assigned with a new one. For example, in a WHSN shown in Figure 10, when a dashed $Node_{19}$ is moved to $Node_{19}$, it will broadcast and rejoin the network with a new parent as root C. Its address will stay as 19 while only its LID changes from 399 to 19 according to its parent's LID. After registering its new LID, it is ready to work at its new position.

*5.2. Responding to Backbone Node Failure.* Energy exhaustion, hardware problem, or unpredictable moving of backbone nodes can cause network failure and routing paths broken. What is worse, with the hierarchical architecture of the WHSN, all descendant nodes of the failed backbone

```
(1) for each node in WHSN do
(2)    Msgs ← node · receiveInterestMsgs();
(3)    node · buildGradient(node · Cap, Msgs)
(4)    if node · feature = plug-in then
(5)       for Msg ∈ Msgs do
(6)          node · broadcast(Msg);
(7)       end for
(8)    end if
(9) end for
```

Algorithm 6: Routing process of DD-Pi.

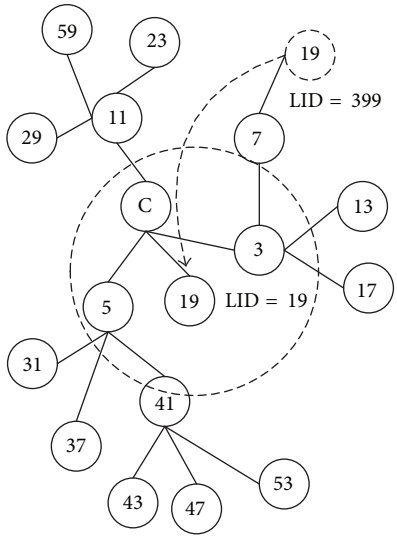



Figure 10: Mobility of D-HiPr.

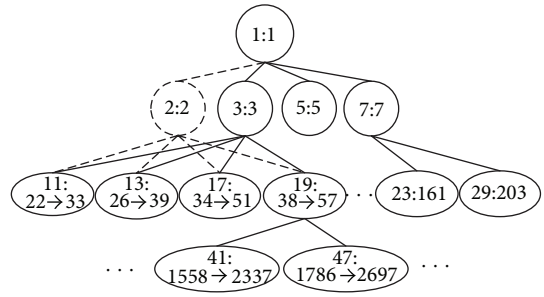



Figure 11: Backbone node failure.

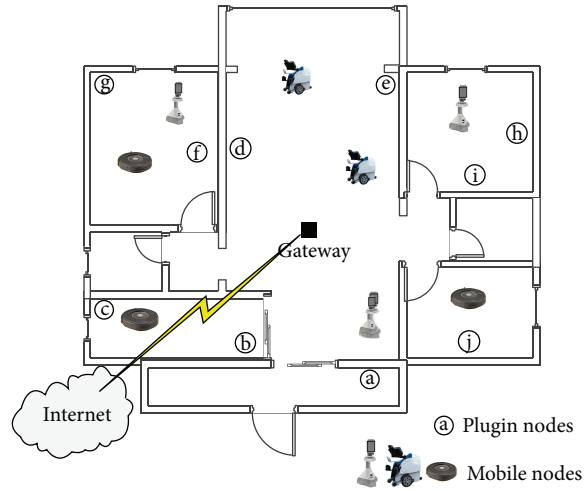



Figure 12: Simulation scenario.

node are disconnected from the network. The cost will be relatively high for all descendant nodes rejoining the network. We propose a mechanism for network restructure that all descendant nodes that are disconnected to their parent node are formed into descendant trees, with the direct child nodes of failed parent node acting as their roots (lines 1-5). Each root of new descendant trees will scan for available parent nodes and select an optimal one from its neighbors (lines 2), while all its descendant nodes keep their associations. Each descendant node gains a new LID from its original parent without sending rejoining messages (lines 7). The gateway will be notified to modify the LIDs for all changes (lines 8). The process for parent node of the descendant trees recovering from failures is described in Algorithm 7.

As presented in Figure 11, each node has a prime address and LID separated by a colon. For instance, $Node_2$ gets failed with its descendant nodes parted into 4 descendant trees: {11}, {13}, {17}, and {19, 41, 47}. Dashed lines denote the links before $Node_2$ gets failed, and arrows in ovals mean the address changes. Node11, Node13, Node17, and Node19 scan for new parent node and resign new LID to their descendant nodes. Each descendant node regains a new LID with inner association links unchanged. This recovery mechanism only involves two procedures. One is that descendant tree root nodes scan for a new parent and the other is that all descendant nodes regain new LID. In contrast with the method that all descendant nodes rejoin the network, it simplifies the network maintenance to avoid all descendant nodes scanning for their parent nodes.

*5.3. Simulation and Results.* In this section, we present our experiments and results to evaluate our approach. The experiments are based on a simulation testbed of a typical household illustrated in Figure 12. There are plenty

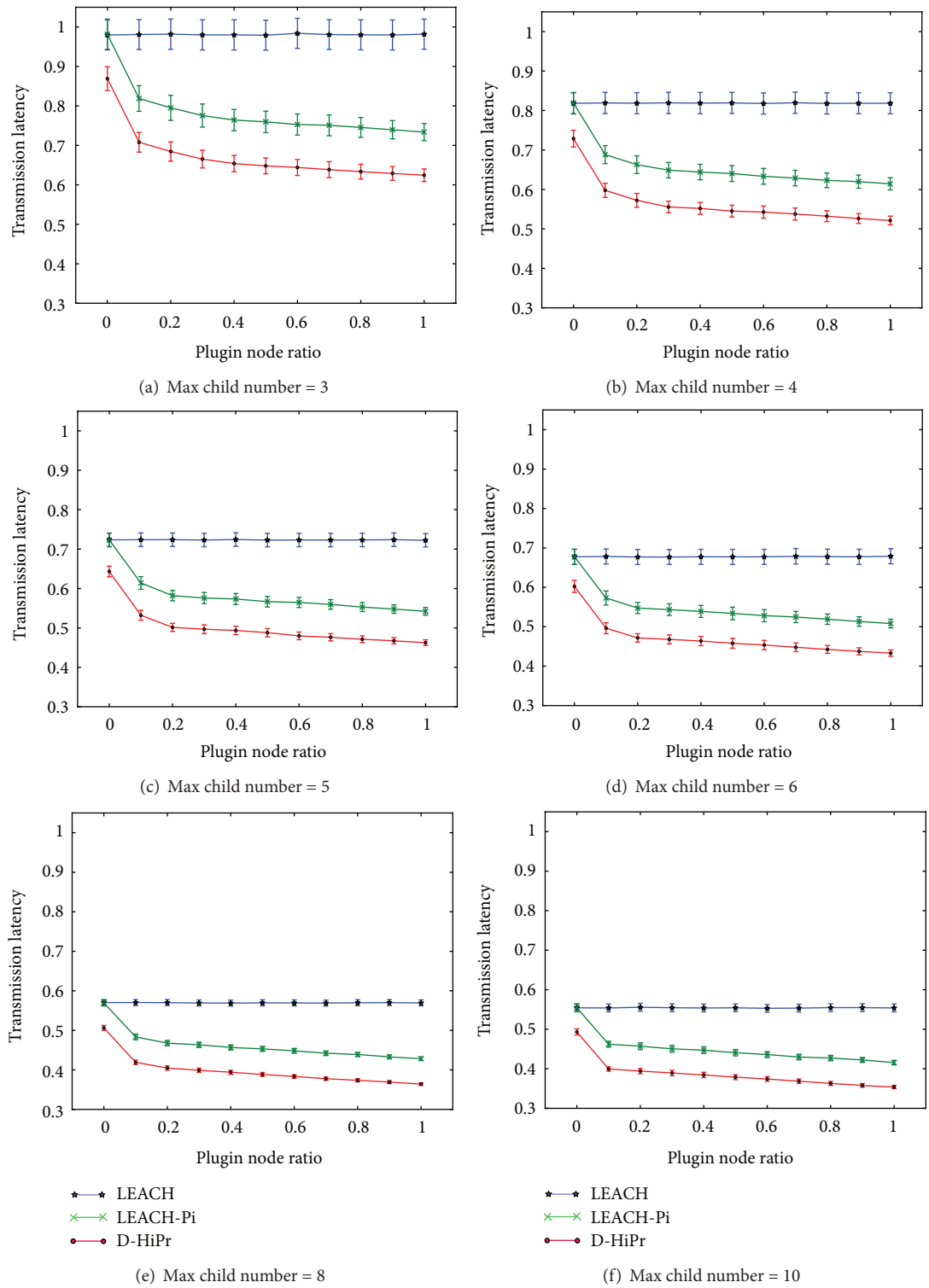
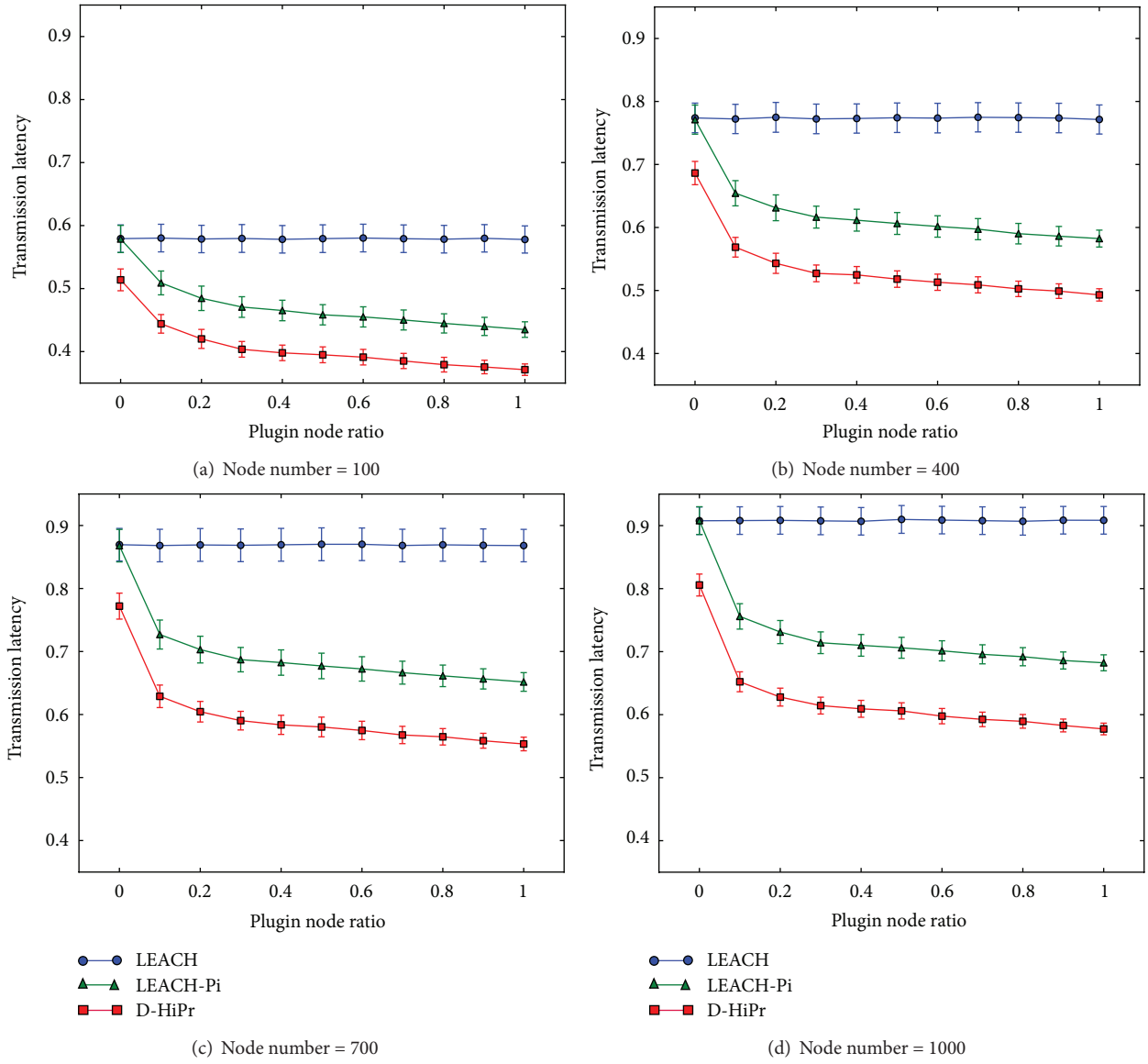



FIGURE 13: Transmission latency by different max child numbers.

of plug-in nodes marked from (a) to (j), such as sensors in the wash machine (a) and in the fridge and the microwave oven in the kitchen (b, c), as well as a bunch of mobile nodes, such as the sensors in household robots and the cleaning machines. Meanwhile, we suppose that the gateway is in the middle of the house. We briefly build three simulations, and in each simulations, based on our hierarchical architecture of WHSN, we testified three algorithms: traditional LEACH, LEACH-Pi, and our proposed D-HiPr algorithm. The simulation results are based on 100 runs.




(a) Node number = 100

(b) Node number = 400

(c) Node number = 700

(d) Node number = 1000

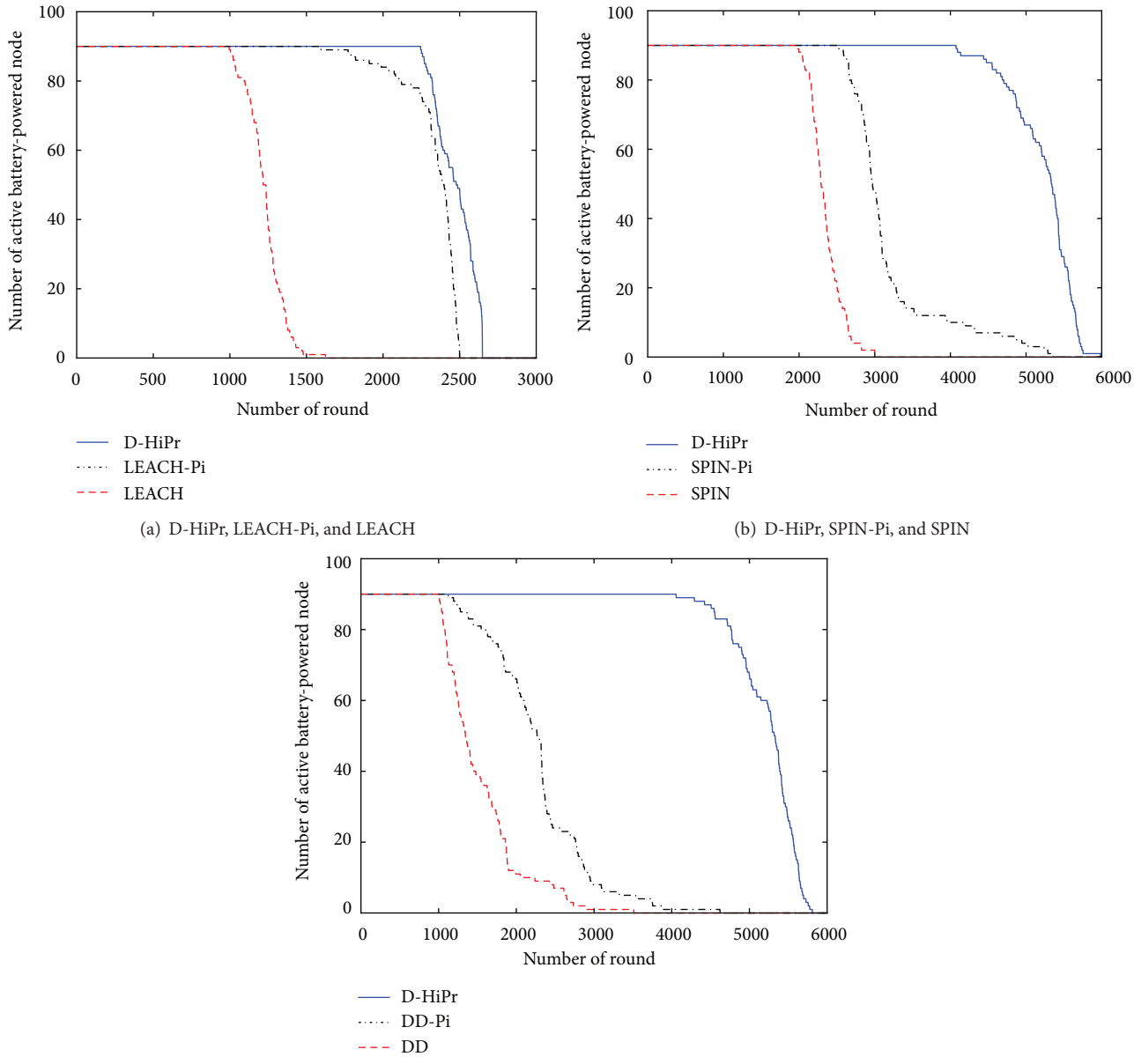
Figure 14: Transmission latencies with different scales of WHSN network.

In the first simulation, we set up a WHSN with 500 of sensor nodes randomly scattered in the household. Although the WHSN always includes two types of sensors, in our experiments, we vary the ratio of plug-in nodes in the network from 0% to 100% as shown in $x$-axes and testify the average transmission latencies of given 500 packets (shown in $y$-axes). In this simulation, we have made six experiments, and in each experiment, we set different numbers of maximum allowed children for each node as 3, 4, 5, 6, 8, and 10. The experiment results are illustrated in Figure 13. As we expected, no matter what the settings are, our D-HiPr algorithm, by taking the advantages of hierarchical architecture, plug-in nodes as backbone, and prime-numbered addressing, takes the least transmission latency than the other two algorithms, while LEACH-PI takes the advantage of using plug-in nodes as cluster head which works better than LEACH. We can also see another evidence in the figure that when there are more and more plug-in nodes, D-HiPr and LEACH-PI are benefited with lower transmission latency, while LEACH stays the same. From all six graphs, when the number of maximum children increases and the depth of the tree decreases, all the three algorithms are benefited with lower transmission latency. The reason is that the average distances between nodes are decreased, but all our previous conclusions hold.

In the second simulation, we setup our WHSN with 100, 400, 700, and 1000 sensor nodes, respectively. By varying the ratio of plug-in nodes in the network from 0% to 100% ($x$-axes), we testify the average transmission latencies ($y$-axes), which is consistent with the first simulation. From the experiment results shown in Figure 14, our conclusions from the last simulation that D-HiPr takes the most advantages and LEACH-PI takes the advantage of plug-in nodes hold. Moreover, when WHSN scales up, the advantage of D-HiPr and LEACH-PI becomes more and more prominent.

```
(1) for node ∈ failedNode · getChildren() do
(2)     node · parent ← FindNewParent(node);
(3)     node · updateLIDFromParent();
(4)     node · notifyGateway();
(5) end for
(6) for child ∈ node do
(7)     child · updateLIDFromParent();
(8)     node · notifyGateway();
(9) end for
```

Algorithm 7: Path Recovery Algorithm.




(a) D-HiPr, LEACH-Pi, and LEACH

(b) D-HiPr, SPIN-Pi, and SPIN

(c) D-HiPr, DD-Pi, and DD

Figure 15: Network sustain lifetime with different protocols.

In the last simulation, we evaluate our design by comparing the sustain lifetimes of the WHSN when it is applied with different protocols explained in this paper. In this simulation, we set the WHSN with 10 plug-in nodes as shown in the simulation scenario and 90 battery-powered nodes randomly scattered in the house. In each time step, several nodes in the network are chosen to send packets to the gateway. For each packet transmission, if a battery-powered node is involved, it will consume some power for communication. If more battery-powered nodes are involved in packet transmissions, the network will be out of power more quickly. We will record in the simulation how long the battery-powered nodes can keep alive.

The experiment results are briefly shown in Figure 15, and in each graph, we use three different algorithms to be compared. Figure 15(a) shows the sustain lifetime of the network with D-HiPr, LEACH, and LEACH-Pi protocols. As we can see, the LEACH protocol which does not consider the strength of plug-in nodes discharges the battery-powered nodes very quickly and halts the network at about 1300 rounds. In contrast, the LEACH-Pi by taking the advantage of plug-in nodes can survive until more than 2000 rounds, while our D-HiPr protocol by integrating all the features of WHSN works well until more than 2300 rounds.

In Figure 15(b), we use the protocols of D-HiPr, SPIN, and SPIN-Pi to test the network, while in Figure 15(c), we use the D-HiPr, DD, and DD-Pi protocols. Similar to the conclusion drawn from Figure 15(a), because the SPIN and DD do not consider the plug-in nodes, they perform very poorly. SPIN-Pi and DD-Pi by considering the merit of plug-in nodes, they perform better. D-HiPr by taking the advantages of plug-in nodes as backbone and prime-numbered routing algorithm, it performs best, and as Figure 15(c) shows, its sustained lifetime can be triple as that of the DD protocol.

## 6. Conclusion and Future Works

In this paper, we have designed a hierarchical architecture for smart WHSN. By classifying sensor nodes into two categories, we can take the advantages of using fixed plug-in nodes as backbone nodes for data transportation while the mobile nodes as leaves to keep the flexibility of the network. Based on our architecture design, we have made three major contributions: a novel prime-numbered hierarchical address allocation and routing protocol; an improvement of classic routing protocols for smart WHSN as well as improved network maintenance algorithms to model node mobility and backbone failure. Our simulation results manifest the efficiency and feasibility of our design. However, even if we are capable of dealing with some of the challenges of the domain, we leave many of the others for the future work. Firstly, our design should be integrated with the physical layer controlling protocol such as 802.15.4, but it is not tested. Second, our design should be verified in some real domains applications of WHSN, which will be an imperative work for the next step. Thirdly, connecting with Internet and matching our design with IPv6 network will be good issue left to the future work.

## Acknowledgments

This research has been sponsored in part by the National Natural Science Foundation of China no. 60905042, National Key Technology R&D Program of China no. 2012BAI22B05, and Huawei Technology Foundation no. YBNW2010086.

## References

[1] Z. S. Wu, "Convergence framework of internet and WSN," Tech. Rep., 2011.

[2] H. Dohler, "Routing requirements for urban low-power and lossy networks," RFC 5488, 2009.

[3] A. Azim and M. M. Islam, "Hybrid LEACH: a relay node based low energy adaptive clustering hierarchy for wireless sensor networks," in *Proceedings of the IEEE 9th Malaysia International Conference on Communications*, December 2009.

[4] W. B. Heinzelman, P. Anantha, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

[5] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proceedings of the 16th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '97)*, pp. 1405–1413, April 1997.

[6] L. H. Zhang, L. P. Zhang, and R. H. Huang, "Energy efficient passive clustering based directed diffusion protocol," in *Proceedings of the International Conference on Advanced Measurement and Test*, 2010.

[7] N. M. White and J. E. Brignell, "Sensors in adaptronics," in *UNSPECIFIED Adaptronics and Smart Structures*, Springer, Berlin, Germany, 2007.

[8] J. Li, Y. Guo, and G. Poulton, "Critical damage reporting in intelligent sensor networks," in *Proceedings of the 17th Australian Joint Conference on Artificial Intelligence*, pp. 26–38, December 2004.

[9] Y. X. Li and X. J. Huang, "The simulation of independent rayleigh faders," *IEEE Transactions on Communications*, vol. 50, no. 9, pp. 1503–1514, 2002.

[10] Z. Shelby, S. Chakravarti, and E. Nordmark, "Neighbor discoveryoptimization for low-power and lossy networks," Tech. Rep., June 2010.

[11] J. P. Vasseur and A. Dunkels, *Interconnection Smart Objects with IP: The Next Internet*, Elsevier, New York, NY, USA, 2010.

[12] M. Esler, J. Hightower, T. Anderson, and G. Borriello, "Next century challenges: data-centric networking for invisible computing," in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '97)*, The Protolano Project at the University of Washington, 2000.

[13] G. K. Ee, C. K. Ng, N. K. Noordin, and B. M. Ali, "Path recovery mechanism in 6LoWPAN routing," in *Proceedings of the International Conference on Computer and Communication Engineering (ICCCE '10)*, May 2010.

[14] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Journal of Ad Hoc Networks*, vol. 3, no. 3, pp. 325–349, 2005.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 937462, 12 pages
doi:10.1155/2012/937462

*Research Article*

# Node Classification Based on Functionality in Energy-Efficient and Reliable Wireless Sensor Networks

**Ning Sun,[1] Youngbuk Cho,[2] and Sangho Lee[2]**

[1] *Department of Computer Science, Chungbuk National University, 52 Naesudong-ro, Cheongju 361-763, Republic of Korea*
[2] *Department of Software Engineering, Chungbuk National University, 52 Naesudong-ro, Cheongju 361-763, Republic of Korea*

Correspondence should be addressed to Sangho Lee, shlee@cbnu.ac.kr

Received 27 May 2012; Revised 24 September 2012; Accepted 30 November 2012

Academic Editor: Regina B. Araujo



Energy efficiency and reliability are two important factors when we design a wireless sensor network (WSN). In this paper, we proposed an energy-efficient and reliable hierarchical mechanism for WSNs. We utilized clustering structure to decrease energy consumption. Because cluster head plays an important role in WSN, we designed a substitute of cluster head (SCH) in order to guarantee the data propagation reliability. Based on different functionalities of sensor nodes we creatively classify sensor nodes into five categories: branch node (BN), cluster head (CH), substitute of cluster head (SCH), intermedium node (IN), and normal node. By adopting different MAC and network strategies for different node categories, energy consumption can be greatly reduced and data reliability can be significantly increased in the proposed WSN mechanism. The simulation results showed that this proposed mechanism realized the objective in aspects of energy efficiency and reliability.

## 1. Introduction

Wireless sensor networks consist of hundreds to thousands of sensor nodes. These sensor nodes are usually homogenous and one or more resource-rich base stations exist in WSNs which is the medium between WSNs and the outside information systems. The sensor nodes collect interest data through sensing the environment parameters such as temperature, wind speed, and humidity, then propagate the sensed data to the base station for furthering processing. Nowadays, WSN technology has been considered mature enough for real-life application such as control and automation, environment monitoring, healthcare, security and surveillance, smart home and building, vehicle tracking and detection, and precision agriculture.

Due to the small size and low price, most sensor nodes have constrains in processors, memory size, communications, and energy capability. Among these limitations, energy saving is always a remarkable issue and has got much attention for designing the WSN protocol. Amounts of protocols in MAC and network layers have been proposed with the purpose of saving energy [1, 2]. Meanwhile, strict energy constraints restrict long and reliable performance of sensor nodes. Hardware often cannot work regularly due to energy depletion. At the same time, sensor networks are often deployed in harsh and hazardous environments, which affect normal operation of sensor nodes. The wireless radios of sensor nodes are severely affected by these environment factors. Communication faults often occur in WSNs due to the interferences. Therefore, reliability is another highlighted challenge for the real-life applications in WSNs. It is necessary for the WSN system to maintain high fault tolerant capability.

Many literatures [2–5] have shown that hierarchical structure is an efficient means to save energy in WSNs. Hierarchical structure can greatly reduce the amount of direct communications between sensor nodes and base station. It is concluded [3] that the energy consumption of cluster head is much higher than normal nodes due to the extra works. By electing cluster heads in rotation the energy consumption can be balanced among nodes and the total energy consumption can be significantly reduced such that network lifetime is prolonged. In this paper we utilized the clustering hierarchical structure.

Besides the hierarchical structure, energy-efficient MAC protocols [1] also achieve energy savings by controlling the radio wake/sleep and arranging reasonable access schedule

to reduce energy waste. MAC protocols for WSNs can be categorized into two main groups: schedule-based and contention-based protocols. In our proposed mechanism, for different category of nodes and in different phases we utilized different MAC protocols flexibly to reduce the energy consumption. By turning off radio of sensor nodes when they are idle in WSNs, our proposed mechanism maximized energy saving.

Although cluster head plays an important role in hierarchical WSNs, most literatures did not consider the reliability of cluster head. Cluster head has to keep awake in the whole process, from cluster formation, data collection, and data aggregation to data propagation. It needs to receive/transmit data all the time and execute complex operations such as TDMA schedule generation and data aggregation. Extra works make it more easily energy exhausted. Long time wakeup also makes it possible to have hardware or communication malfunction. Once cluster head fails, the cluster working has to halt and whole cluster's data would be lost. So the reliability of cluster head is very important. In this paper, we proposed an approach in which a substitute of cluster head (SCH) is selected and works in the sudden failure of cluster head. For substitute of cluster head, it needs more energy consumption than other noncluster head nodes. But by sacrificing consumption in one node, the reliability of whole cluster could be increased. Moreover, by selecting SCH in rotation, the average energy consumption is insignificant and does not affect the whole system.

Finding multiple node-disjoint paths also provides fault tolerance in routing. The system can switch from an unavailable path with broken links to alternative candidate paths. Major solutions [6, 7] for fault-tolerant and reliable data propagation of WSNs are based on the multipath routing paradigm, which provides each sensor node with alternative paths to destination. It solved the problem that wireless communication is not stable and provided fault tolerance for data transmission. In the phase of aggregated data transmission from cluster heads to the base station, we perform multipath discovery algorithm for increasing reliability.

Based on the above strategies, we classify sensor nodes into five categories according to their different roles in WSNs. They are branch node (BN), cluster head (CH), substitute of cluster head (SCH), inter-medium node (IN), and normal node. For each category, it executes alternative network and MAC protocol and operates specific functions, such as schedule generation, data aggregation, energy alarm, radio control, route discovery, and data transfer. All the design standards for each category are based on considerations of energy efficiency and reliability.

There are some original and progressive contributions in this paper: firstly, we proposed an energy-efficient and reliable WSN mechanism, which adopted multiple strategies including the hierarchical clustering structure, multipath routing discovery protocol, energy-efficient and flexible MAC protocol, and substitute of cluster head. Secondly, we classify sensor nodes into five categories according to their functionalities in WSNs. For each category, we considered extra works that are responsible for and designed flexible MAC protocol. This strategy guarantees more fairness in node energy consumption and prolongs the network lifetime as far as possible. Thirdly, the mechanism of substitute of cluster head (SCH) accounts for the situation of CH's sudden failure. Moreover, multipath discovery scheme can solve the problem when a path fails. These guarantee the reliability and robustness of the system. Fourthly, all the phases including cluster formation, data transmission within cluster, and multiple path transmission are in distributed manner; it guarantees the scalability of the network. Finally, we use two global parameters of total energy and node sum to improve the performances of both cluster head/SCH election and routing discovery. In our scheme, these two parameters are updated after every round and calculated by the base station. It is an efficient way to control the network quality.

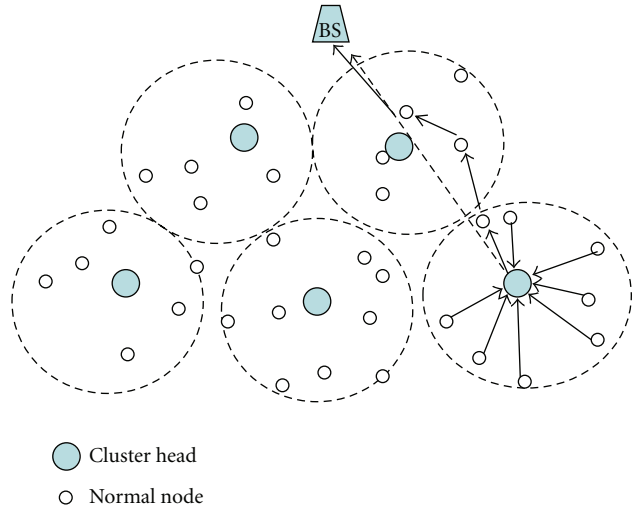



Figure 1: Clustering structure in WSNs.

The remainder of this paper is organized as follows. Section 2 presents the related works. Section 3 describes the classification of sensor nodes and explains the detailed algorithms of each category. Section 4 analyzes the simulation results and shows why this mechanism outperforms others. Finally, Section 5 draws the conclusion.

## 2. Related Works

### *2.1. Energy Saving Mechanisms*

*2.1.1. Hierarchical Structure.* The aim of the hierarchical structure is to reduce the energy consumption by reducing the amount of direct communications. Clustering structure is the main form of hierarchical structures in WSNs.

The literature [2, 3] stated a survey of energy-efficient hierarchical cluster-based protocols in WSNs. As illustrated in Figure 1, the network is divided into a few clusters. Nodes are grouped into clusters with a cluster head responsible of routing from the cluster to the base station. In this model, data is firstly collected periodically by normal sensor nodes. Instead of directly transmitting data to the base station, cluster heads collect the data from the sensor nodes within their clusters and send an aggregated data to the base

station directly or through a multihop path (based on the communication radio range of CHs and scale of network).

Cluster heads consume relatively more energy than none-cluster head nodes because cluster heads have more loads to handle. Hence, cluster heads may run out of their energy faster than other sensor nodes.

LEACH [4] is the classical clustering hierarchical protocol for WSNs, which incorporates randomized rotation of the high-energy cluster head position among the sensors to avoid draining the energy of any one sensor in the network. In this way, the energy load of being a cluster head is evenly distributed among the nodes. The probability of cluster head election depends on the number of clusters $k$ and the sum of the sensor nodes $N$:

$$P_i(t) = \begin{cases} \dfrac{k}{N - k * (r \mod (N/K))} : & C_i(t) = 1. \\ 0. \end{cases} \quad (1)$$

In (1), $r$ is the round number and $C_i(t)$ depicts whether node $i$ has been the cluster head in recent rounds.

LEACH enhances network lifetime and energy consumption compared with the flat algorithms. LEACH approach is based on the assumption that all nodes start with an equal amount of energy and cluster heads have uniform energy consumption. Obviously the latter is not realistic. One shortcoming of LEACH is that it does not consider the energy factor when electing cluster headers, whereas we designed a more efficient cluster head election algorithm. Secondly, LEACH does not work well in the dynamic network. It is because parameters (number of clusters, number of nodes) are programmed into the nodes a priori, but in reality these parameters continuously change with the depletion of nodes. Furthermore, LEACH is not scalable for large size of WSN. It is assumed that every node has enough power to communicate with others and BS directly, but it is not realistic for a large scale of WSN.

The subsequent LEACH-optimized protocols in literatures [5, 8, 9] mostly improve the capability of cluster head distribution, so that the energy consumption of the whole network is reduced and the system lifetime is prolonged. However, most of these hierarchical protocols only improved the algorithm of cluster formation and gave little consideration on aggregated data transmission. Since the aggregated data is important, the reliability of transmission should be guaranteed.

*2.1.2. Energy-Efficient MAC Protocols.* In wireless communications, energy waste mainly comes from the following sources: idle listening, data collision, overhear and control packet overhead, and so forth. Among them, idle listening causes nearly half of the energy waste in WSNs [1], so turning off the radio in unnecessary time is almost the most efficient way to save energy. Energy-efficient MAC protocols achieve energy savings by controlling the radio wake/sleep and arranging reasonable access schedule to reduce energy waste caused by these sources.

MAC-layer protocols can be categorized into two types: schedule-based and random access protocols. The former is the protocol in which access to the channel is based on a schedule. Channel access is usually limited to one node at a time. Random access protocols avoid preallocation of resources to nodes, so they are more flexible than schedule-based protocols. However, this results in collision. The main objective of random access protocols is to minimize the occurrence of collisions.

TDMA (time-division multiple access) is a typical schedule-based MAC solution. It is achieved by dividing the radio frequency into time slots and then allocating unique time slots to each communicating node. Nodes take turns wake up and transmit/receive data. In LEACH protocol, it uses TDMA in data transmission within cluster. The time line of LEACH protocol is as shown in Figure 5(a). Most clustering WSN protocols use TDMA as the MAC protocol to manage the access of nodes within a cluster.

Although TDMA protocol can reduce energy consumption greatly, it limits the number of joint nodes and the constraints of time allocation so that it is short of scalability and flexibility. Contention-based protocols require no coordination among the nodes accessing the channel and provide more scalability and flexibility. CSMA series are the popular contention-based protocols. For WSNs, S-MAC [10] is proposed to reduce the energy efficiency by using a low duty cycle operation. It also provided schemes for collision avoidance, overhearing avoidance, and synchronization. But the energy saving may result in high latency and low throughput. In the same time, collisions cannot be avoided in content-based MAC protocols. In our proposed mechanism, we use both TDMA and content-based MAC protocols in different phases.

### 2.2. Reliability Mechanisms

*2.2.1. Multipath Routing Algorithm.* As described in [6], there has been a lot of research works in multipath routing for WSNs in recent years. Using multipath routing provides fault tolerance of node failure along single path and increases the network reliability. Constructing node disjoint paths has been considered the most reliable method in WSNs.

H-SPREAD [7] uses the branch-aware flooding method to find a set of node-disjoint paths. As the extension of the flooding-based beaconing protocol [11], H-SPREAD proposed an $N$-to-1 multipath discovery protocol. The multipath discovery procedure is presented in two phases, with each phase implementing one of the mechanisms. The mechanism used in the phase one, which is termed as a branch-aware flooding, takes advantage of the simple flooding technique. Without introducing additional routing messages, the mechanism is able to find a certain number of node-disjoint paths, depending on the density of the network topology. The mechanism used in the phase two, which is termed as a multipath extension of flooding, helps to exchange the node-disjoint paths found in the phase one among the nodes on different branches. At the cost of some more message exchanges, it is able to increase the number of paths found at each sensor node.

Figure 2 shows the algorithm of branch-aware flooding. We name the one-hop neighbors of the BS as branch

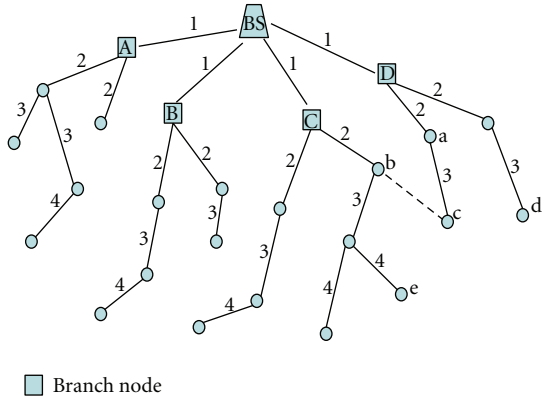



Figure 2: Branch-aware flooding algorithm.

node (BN). If a node receives a request message from the neighbor with different BN value, it can be regarded as the discovery of an alternative path. For example the square nodes (A, B, C, D) represent BNs. Node c firstly receives message from node a, then it constructs a primary path (A-a-c). And then c continuously receives messages from nodes b, d, e. When judging an alternative disjoint path, c will compare the BN of each sender. If BN of sender is the same with its, it means the new path is joint; otherwise, the new path is a disjoint one. For the paths with the same BN, c will compare the hop number. The less hop number path is selected. By this way, an alternative disjoint path (C-b-c) is constructed by c.

In our proposed path discovery algorithm, similar routing protocol is used. Instead of store alternative paths before data propagation, we utilized a more flexible path discovery algorithm.

Branch node and nodes nearer, the base station relay more packets and more actively participate in communication. As a result, these nodes expand more energy and are more failure prone due to energy depletion. How to conserve energy in these nodes is an important issue, for if these nodes are exhausted, all paths to the base station are broken. In previous multipath routing protocols for WSNs [6, 7, 12, 13], the problem of how to avoid extra energy consumption in neighbors of base station has not be discussed, whereas in our proposed mechanism this problem was considered.

*2.2.2. Clustering Fault Tolerant Mechanism.* In [14, 15], fault tolerance of cluster-based WSNs is discussed.

CPEQ in [14] is a cluster-based routing protocol. It takes account of the reliability of data transmission within clusters. It assumes that data propagation within clusters is multihop. It implements a path repair mechanism within clusters. It did not account for the situation that cluster head failed. The literature [15] uses the CHs coordination mechanism for fault tolerance. CHs exchange status message after every cycle to detect the failure of CH. If failure is detected, the nodes with the fault CH are assigned to near cluster in next round automatically. This mechanism needs CHs exchange a lot of messages resulting in much overhead. Moreover, it causes node isolation problem after reassignment.

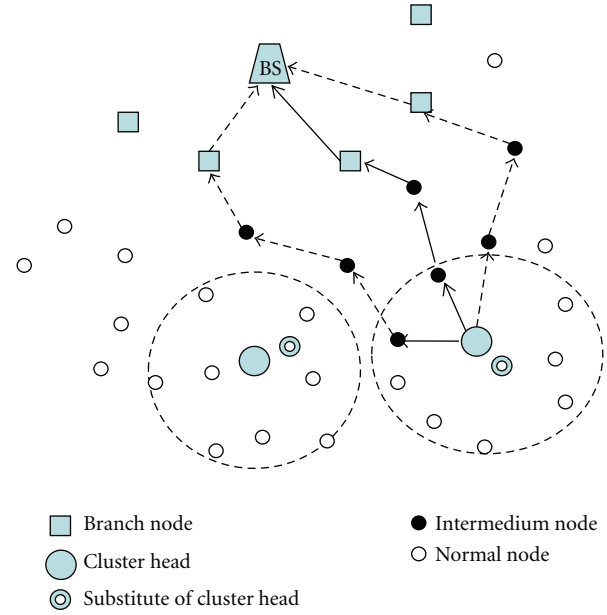



Figure 3: Node classification mechanism.

## 3. Node Classification Mechanism

*3.1. Network Model.* We assume this proposed mechanism is for a large scale and dense wireless sensor network. One or more base stations exist as the gateway between WSN and the outside information system. Except base station, all sensor nodes have similar initial energy and capabilities (sensing/processing/communication). Nodes need not change their radio range in our system. Communications between nodes are bidirectional and symmetric. All nodes are able to be aware of their residual energy. Each node has a unique identifier to be marked in the network. Within a cluster the communication from normal nodes to CH is one hop, whereas the communication from cluster heads to base station is multihop. This approach is more suitable for the applications in which sensed data is collected periodically. Finally, synchronization is necessary among sensor nodes due to the demand of MAC protocol.

*3.2. Energy-Efficient and Reliable Mechanism.* As in [13], this protocol mainly consists of three phase. The first phase is the initialization phase, in which each node retrieves the information about the neighbors and the network. The second phase is the cluster formation phase, in which clusters are formed in a distributed manner. The final phase is the data propagation phase, which includes two steps: sensed data is transmitted within clusters, and aggregated data is transmitted from cluster heads to base station. Our scheme utilized hybrid MAC protocols for different phases. The entire WSN is displayed in Figure 3, where we can see different categories of sensor nodes.

*3.2.1. Phase One: Initial Broadcasting.* This phase starts from the BS broadcasting a request message. The format of this message is {INI, RID, BID, SID, $E_{re}$, $H_{count}$, $E_{to}$, $N$, $T_1$}, where

Table 1: Structure of neighbor information table.

| Neighbor ID | IsParent | BID | Residual energy | Hop count | RSSI |
|---|---|---|---|---|---|

INI indicates that the type of message is initialization; RID is the round identifier, which is generated by the base station; BID is the identifier of the branch, that is, identifier of the branch node; SID is the identifier of the sender node; $E_{re}$ is the residual energy of the sender node; $H_{count}$ is the hop count from the sender to the base station; $E_{to}$ depicts the total energy of all nodes, and it is calculated by the base station. Finally $N$ is the sum of nodes after last round. Here $E_{to}$ and $N$ are prepared for the selection of cluster heads. $T_1$ depicts the timestamp for the ending of phase 1 and the beginning of the next phase.

Hop by hop, node receives the broadcast message, updates information of message, and rebroadcasts it. Upon receiving the first message, node marks the sender node as its parent. After this phase, every node decides whether it is a Branch node, stores the parameters of total energy $E_{to}$ and total number of nodes $N$ for the next phase, and constructs the table of neighborhood information, which will be used to discover multiple paths in the data propagation phase. This mechanism takes advantage of the broadcast nature of the wireless communication. Each node rebroadcasts once and only once. The table of neighbor information is as shown in Table 1.

In Table 1, node stores all the received neighbors' information, including neighbor node ID, whether it is the parent node, branch ID of this neighbor, residual energy, hop count to BS, and the received signal strengthen indicator (RSSI) which is proportional to the distance between them. In this phase, branch aware flooding algorithm is used. In this process even some messages were lost due to some node failures or link conflicts, and nodes could obtain the network parameters from other neighbors and construct the approximate impress about the neighbors. All sensor nodes keep awake until this phase ends, that is, time $T_1$.

*3.2.2. Phase Two: Cluster Formation.* After previous phase, every node has got the information about the total energy $E_{to}$ and sum of nodes $N$. After time $T_1$, phase two begins and each node begins to decide whether to be a cluster head in a distributed manner.

Once the node has selected itself to be cluster head, it broadcasts an advertisement message (ADV) using CSMA MAC protocol. This message is a small message containing the node's ID, cluster's ID, and an ADV header. Nodes decide which cluster they should belong to by choosing the cluster head that requires the minimum communication energy, based on the received signal strength of the advertisement messages from cluster heads. After each node has decided to which cluster it belongs, it transmits a join-request message (JOIN-REQ) back to the chosen cluster head also using CSMA MAC protocol. This message consists of the node's ID, the joining cluster head's ID, the residual energy, and the hop count to BS of the node.

The cluster head node sets up a TDMA schedule and broadcasts this schedule to the nodes in the cluster. In the same time, according to the parameters of residual energy and hop count and the RSSI, CH selects a node as a SCH. SCH declaration is broadcasted in the same message with the TDMA schedule. Node matching the SCH_ID will mark itself as a SCH.

In this phase, CSMA MAC protocol is utilized to avoid communication collisions. To conserve energy, branch nodes do not join any cluster, so they turn into asleep in the whole phase. Other sensor nodes keep awake during cluster formation.

*3.2.3. Phase Three: Data Transmission.* This phase mainly consists of two steps: at first the data propagation within a cluster, then the data propagation from the CHs to the BS, which is multihop transmission along multipaths.

In the data transmission within clusters, nodes comply with the TDMA MAC protocol. Nodes send their data to the cluster head at most once per frame during their allocated transmission slot. Once the cluster head receives all the data, it performs data aggregation to enhance the common signal and reduce the uncorrelated noise among the signals. In our proposed mechanism, cluster heads execute a spatial correlation on collected data. In our approach, after every round, BS needs to know the whole residual energy of all nodes and the sum of nodes alive. Therefore, in this step, besides the data aggregation, the cluster head also does the following computations:

$$N_{\text{cluster}} = \sum_{i=1}^{n} i,$$
$$E_{\text{cluster}} = \sum_{i=1}^{n} E_i - N_{\text{cluster}} * (E_{\text{TX}} + E_{\text{RV}}) - E_{\text{fusion}}. \quad (2)$$

$N_{\text{cluster}}$ is the sum of nodes in this cluster, while $E_{\text{cluster}}$ is the whole residual energy value of all nodes in this cluster, which accounts for the energy consumption during the data propagation. Finally, the cluster head sends the data message, which consists of the message type (DATA), aggregated data, $N_{\text{cluster}}$, and $E_{\text{cluster}}$.

During the above process, if the cluster head fails, for example, running out of energy, all the sensed data within the cluster will be lost. It is why we select a substitute of cluster head. Even if CH cannot work normally, SCH will take over the works of CH to guarantee the reliability of data propagation.

In this step, CHs and SCHs keep awake during the whole process, whereas other nodes only wake up in its allocated time slot.

The second step is the aggregated data transmission from the cluster heads to the BS. Since the BS may be far away, this is a multihop and high-energy transmission. Before data transmission there is a process that we call "INs Broadcasting" as shown in Figure 4. In this process, CHs (or SCHs) firstly select appropriate next hop nodes based on the information of neighbor table and broadcast a NEXT_IDs message. This message not only includes the

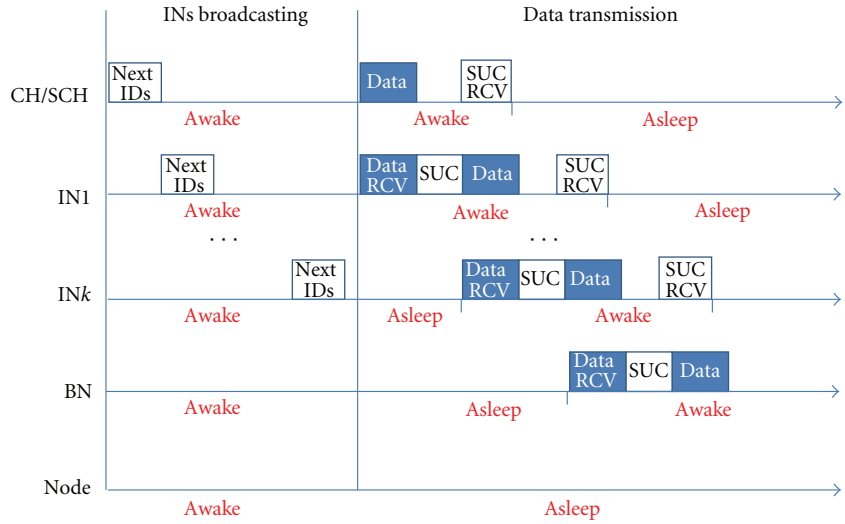



Figure 4: Timeline for each category during data propagation from CHs to BS.

next hop nodes' ID, but also includes hop count from CH to BS along each next hop node. The nodes which receive NEXT_IDs message from CHs/SCHs will mark themselves as IN1 select their next hop nodes, and rebroadcast NEXT_IDs messages. NEXT_IDS messages recurrently broadcast until they are received by branch nodes. By this way, sensor nodes in WSNs can be distinguished between intermedium nodes and none-intermedium nodes.

In the data transmission from CHs to BS, only CHs/SCHs, INs, and BNs keep awake and take part in the data transmission, whereas other nodes turn into sleep until the process ends. As shown in Figure 4, after every data transmission, the sender node will wait for the SUC message from the recipient. If the SUC message is received, the sender will turn into sleep until the end of data transmission; otherwise, if the sender cannot receive the SUC message in a limited period of time, the sender will resend the data to another next hop node and the data transmission process will repeat. Furthermore, in last INs broadcasting process, each IN knows the hop count of the path it resides, so it can estimate the waiting time $T_{\mathrm{IN}i}$ according to (3):

$$T_{\mathrm{IN}i} = k\left[\left(H_{\mathrm{path}} - H_{\mathrm{IN}i}\right) \times \Delta T\right]. \tag{3}$$

In (3), $H_{\mathrm{path}}$ is the whole hop count of the path, whereas $H_{\mathrm{IN}i}$ is the hop count from IN$i$ to BS. $\Delta T$ is the approximate time for data transmission between two one-hop nodes, which includes the time for data transmission and receiving, plus the time for SUC message transmission and receiving. Coefficient $k$ is larger than 1, which accounts for delay of data transmission. During period of $T_{\mathrm{IN}i}$, INs keep asleep and do not wake up until the end of $T_{\mathrm{IN}i}$. Using this flexible MAC protocol, all nodes turn into sleep in their unnecessary time, so energy consumption in this phase could be decreased greatly.

There are several disjoint paths for data delivery from CHs to BS. As illustrated in Figure 3, the solid arrow path is the primary path from CH to BS, whereas the dashed arrow paths are alternative paths as the backup of primary path.

Ultimately, the sensed data and the parameters of the network (sum of nodes $N$, whole residual energy $E_{\mathrm{to}}$) should be retrieved by the BS. Then the BS initiates another round.

*3.3. Node Classification.* Based on the functionality in WSN, we defined five classifications for sensor nodes. It is remarkable that some category may exist only in a specific phase not the entire process.

*3.3.1. Branch Node (BN).* The definition of the BN is the one-hop neighbors of the base station. According to the branch aware flooding algorithm [7], each BN represents one branch. As shown in Figure 2, nodes A, B, C, and D are BNs. In phase 1, upon receiving the first message, the node marks the sender node as its parent and the path in which parent node resides is regarded the primary path. If a node receives an initial broadcasting message with different BN value from the value of its parent node, it can be regarded as the discovery of an alternative path.

According to the simulation result in [7], the nearer to the BS, the node has more burdens on data transmission. The BN acts a critical role in the network, because once it is exhausted the whole branch is separated and the downstream paths are correspondingly failed. In order to conserve energy, in our proposed mechanism the BN does not join the cluster formation and data sense. It only acts as a router in the network. Furthermore, BNs keep asleep in phase 2, by this way BNs conserve much energy. If its energy is below a limitation value, it should announce that it abandons the role of branch node and transforms to a normal node.

In phase 1, if node $i$ received initial broadcasting message directly from BS, it will decide whether it should be a branch node according to its residual energy:

$$E_{re} \geq E_B = k(M * (E_{\mathrm{TX}} + E_{\mathrm{RC}})). \tag{4}$$

$E_{TX}$ is the energy consumption in data transmission, whereas $E_{RC}$ is in data reception. $M$ is proportional to the number of messages. $k$ is the coefficient. If the condition is satisfied, node $i$ marks itself as the BN. By this strategy, we limited branch nodes to the nodes with more residual energy avoiding early energy exhaustion of some nodes.

*3.3.2. Cluster Head (CH).* In our approach, cluster heads are selected in distribution based on the residual energy, the density, and the distance to base station. CH is in charge of data receive, TDMA schedule generation, data process, data aggregation, and data transmission. The energy consumption of CH is much quicker than normal nodes. So an evenly CH selection algorithm is necessary.

In our paper, CH is selected in a distributed manner. Node $i$ generates a random number $P$ between $(1, 0)$. Then node $i$ calculates the probability $P_i(t)$ as (5). If $P < P_i(t)$, node $i$ will declare itself a CH:

$$P_i(t) = \min\left\{k * \left[\alpha\frac{E_i}{E_{to}} + \beta\frac{N_{neig}}{N} + \gamma\frac{1}{H_{toBS}^2}\right], 1\right\}. \quad (5)$$

$E_i$ is the residual energy of node $i$. $N_{neig}$ is the sum of neighbors of node $i$, which can be retrieved from the neighborhood table. $N$ is the total number of all sensor nodes, whereas $E_{to}$ is the total energy of all nodes. $H_{toBS}$ is the hop number to BS. $k$ is the optimal number of clusters, which can be calculated as about 5% of $N$. $\alpha$, $\beta$, and $\gamma$ are the coefficients, whose values are in the range $(0, 1)$.

Compared with LEACH, the CH selection algorithm considers the residual energy, the density (number of neighbors), and the distance (hop count) to BS of each node, instead of the simple randomized rotation. So the cluster election algorithm has more efficiency. In here two global parameters are used to control the quality of clustering, which are the total number of existing nodes $N$ and the total amount of energy of all nodes $E_{to}$. These two parameters are updated after one round and reflect the status of network.

*3.3.3. Substitute of Cluster Head (SCH).* Substitute of cluster head (SCH) is defined as the substitute when CH suddenly fails. Because of the important role in WSNs, the reliability of cluster head is very important. In this paper, we proposed a method where a substitute of cluster head is selected and works when cluster head has failure. For this substitute of cluster head (SCH), it needs more energy consumption than other noncluster head nodes. But by sacrificing energy consumption in one node, the reliability of whole cluster can be increased. Moreover, by selecting SCH in rotation, the average energy consumption is insignificant and does not affect the whole system. This strategy guarantees that the data could be transmitted correctly even if the cluster head fails. This could improve the reliability and fault tolerance of the system.

The strategy of SCH includes two critical parts: the first is SCH selection algorithm, and the second is the MAC protocol designed for SCH.

SCH selection work is done by CH. After CH is selected, CH broadcasts ADV message, then nodes send their information including their ID, residual energy, and hop count to BS in JOIN-REQ messages to the cluster head. Based on these parameters and received signal strengthen indicator (RSSI), CH finally appoints a node as a substitute who has more residual energy, less hop count, and nearer to CH. SCH appointment is broadcast with the TDMA schedule message:

$$\mathrm{ID(SCH)} = \mathrm{ID}\left(\max\left\{k_1E_i + k_2\mathrm{RSSI}_i + \frac{k_3}{H_i}\right\}\right) \quad (1 \le i \le n). \quad (6)$$

Equation (6) shows the principle to select the SCH. A CH receives JOIN-REQ messages, which include the residual energy and hop count to BS of all cluster member nodes. Through calculation as in (6), the node which has the maximal composite value of residual energy, hop count, and RSSI is selected as the SCH.

During the process of data transmission within cluster, if the cluster head is in any urgent situation, for example, physical fault or communication fault, at this time the data transmission or data aggregation process is carrying on, then all the sensed data within the cluster will be lost. It is why we select a substitute node. In this process, all nodes within the cluster comply with the TDMA schedule except SCH. SCH keeps awake during all the process. It receives data as the CH does. After the TDMA schedule expires, different with LEACH, there is a specific time slot for CH to transmit an RCV_SUC to SCH. In this timeout, if SCH receives RCV_SUC, it empties the memory and waits for the next phase; otherwise it means that there are some problems with CH, so SCH takes over the works of data aggregation and aggregated data transmission. By simply inserting a timeslot, the TDMA schedule need not be altered at all. It is a very reliable and flexible fault tolerant scheme. The difference of our MAC protocol with LEACH is illustrated in Figure 5.

*3.3.4. Intermedium Node (IN).* The data transmission between CHs and BS is multihop communication. In this process, we assume that only nodes in the delivery paths need to be awake. We name these nodes intermedium nodes (INs). IN is specially defined for the process of data transmission from CHs/SCH to BS. By this way, the idle listening and overhearing of nodes not in the routes from CHs/SCHs to BS are avoided, achieving significant energy saving.

Before transmitting data, CH/SCH performs a route discovery phase beforehand. After this phase, CH/SCH determines its next hop nodes and broadcasts a message. The next hop nodes receive the message, mark themselves as Ins, and keep awake in the following data transmission process, whereas other nodes go to sleep until the data transmission ends. The MAC protocol for this process has been discussed in phase 3.

The detailed next hop node discovery algorithm is described in Algorithm 1. At first the CH (or SCH) checks its neighborhood table as shown in Table 1. The node marked with "Parent" is the next hop along the primary path from the cluster head. Then CH continuously looks for the neighbors with different BID value from its parent. After comparing the energy and hop count, the CH chooses

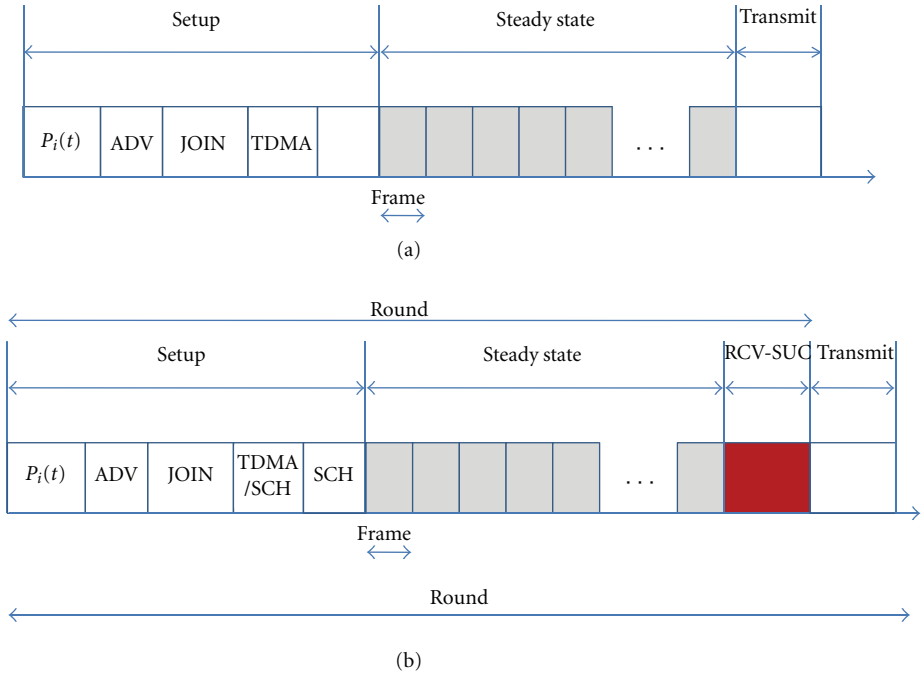



Figure 5: (a) Timeline of LEACH. (b) Timeline of the proposed protocol.

the next hop nodes. Other INs execute the same algorithm recursively to discover the next hop nodes. This route discovery algorithm generates multiple disjoint paths to improve the reliability of data delivery.

*3.3.5. Other Nodes.* The other nodes are in charge of basic works such as initialization, data sensing, cluster formation, and data transmission (within cluster). Normal nodes have minimal energy consumption than other categories.

## 4. Performance Analysis

In this section, we evaluated the performance of our proposed mechanism based on two factors: energy consumption and reliability. At first, we observed the energy dissipation amount of each category in one round and analyzed the cause in details. Then we analyzed the energy savings by using strategies of branch node energy conservation and flexible INs MAC protocol. We also compared the proposed mechanism with previous clustering based works [4, 5] on the performance of network lifetime, which is a synthesis factor to evaluate the superiority of wireless sensor networks. In the next we set different node failure rates and observed the average packet delivery ratios, which reflect the success rate of packet transmissions to the BS. We compared the situations of with/without SCHs and multipath routing algorithm, respectively. WSN reliability can be measured through this result.

*4.1. Experiment Setup.* We have carried out a set of experiments using NS-2 simulator. In the simulation, we used the same radio energy dissipation model as in [4]. The values of parameters used for simulation are as shown in Table 2.

Table 2: Simulation parameters.

| Parameters | Value |
|---|---|
| Topology size | 1000 m × 1000 m |
| Number of nodes | 1000 |
| Transmission range | 50 m |
| Sensing range | 40 m |
| Initial energy | 5 J |
| Packet size | 4000 bits |
| Location of BS | (600, 160) |
| Bandwidth | 1 Mb/s |
| Electronics energy | 50 nJ/bit |
| Free space $\varepsilon_{fs}$ | 10 pJ/bit/m$^2$ |
| Multipath fading $\varepsilon_{mp}$ | 0.0013 pJ/bit/m$^4$ |
| Average data aggregation energy | 5 nJ/bit/singal |
| Percentage of CHs (%) | 3, 4, 5 |

In the following experiments, we set that the percentage of cluster head number 3%, 4%, and 5%, respectively. We evaluated the energy dissipation of different categories of nodes in one round and the reliability with two kinds of strategies. Every result shown is the average of 100~200 experiments, and the confidence interval used in the simulation is 95%.

*4.2. Energy Dissipation.* The first experiment we do is to obtain the average energy dissipation of different category of nodes in one round. Here we divided a round into four phases as illustrated in Figure 6, which is little different from the phases we divided in Section 3.2. We can observe that in each phase the energy dissipation of each kind of node

```
(1) Define:
(2)   n_i: sensor node i
(3)   NB_i: the set of neighbors of n_i
(4)   NEXT_i: the set of next hop nodes of n_i
(5)   IN: Inter-medium Node
(6) Begin
(7)    For each j ∈ NB_i do
(8)      If n_j · IsParent == True then
(9)           n_j → NEXT_i                          //Primary path is found
(10)       Else
(11)         IsAlterPath = True
(12)         For each k ∈ NEXT_i
(13)           If n_j · BID == n_k · BID then
(14)                IsAlterPath = False
(15)                If n_j · H_count < n_k · H_count && n_j · E_re > E_0 then
(16)                     n_j → NEXT_i
(17)                     n_k is removed from NEXT_i
(18)                End If
(19)           End If
(20)         End for
(21)         If IsAlterPath = True then
(22)           n_j → NEXT_i                        // Alternative path is found
(23)         End If
(24)       End If
(25)   End For
(26)   n_j broadcast NEXT_i
(27)   For each k ∈ NEXT_i do
(28)     Receive NEXT_i
(29)     IsIN = True
(30)     Execute 7
(31)   End For
(32) End
```

Algorithm 1

is different. It is due to adopting specific network and MAC protocol for different category in each phase. Now we will analyze the reasons to cause difference of energy dissipation for different categories.

The first phase in a round is the initial broadcasting. In this phase, all sensor nodes keep awake and broadcast/receive messages to collect information of whole network and the neighbors. Energy dissipation in here is the same for each category of nodes. As shown in Figure 6, lines representing varied category of nodes coincide in this phase.

The second phase in a round is the cluster formation phase. In this process, BNs go to sleep and do not join any cluster formation activities in order to conserve energy. CHs consume more energy than other nodes due to their high workload.

Third phase is the data transmission within clusters. BNs continue sleeping, so their energy consumption is 0. Because in this process TDMA MAC protocol is used, normal nodes only wake up in their allocated time slot to transmit sensed data. So the average energy dissipation of normal nodes is very low. Whereas CHs and SCHs keep awake during the whole phase and continuously receive data, this makes their energy consumption increased rapidly.

The last phase is the data transmission from CHs/SCHs to BS. In this phase we utilized a flexible MAC protocol as we have stated in Section 3.2.3. Nodes are divided into INs and normal nodes. Only nodes participating in data transmission keep awake, while other nodes go to sleep after declaring they are not INs. In this phase CHs (or SCHs), Ins, and BNs all wake up during their work time and turn to sleep after work and before the packets arrive. It is shown that BNs consumed maximal energy because nodes nearer to BS have high workload to transfer data.

Overall, CHs greatly have more energy consumption than other nodes in a round. This problem could be solved by electing CHs in rotation. It is suitable to the SCHs too.

It needs to mention that the timeslot for each phase marked in horizontal axis is not the exact one in a round. In reality, the period of data transmission within cluster occupied the most time in a round. In the experiment, we set the percentage of cluster heads is 4%. Because the trend is same for other percentages, it is not necessary to do the alternative experiments.

In this experiment, we can find that BNs greatly conserved energy by limiting their participations only in some specific phases (other time they should be in sleep mode). The amount of energy dissipation of BNs was reduced as little as the normal nodes. We compared mechanisms with BN sleep strategy and without BN strategy in different percentage of CHs. In Figure 7(a), the result shows that the

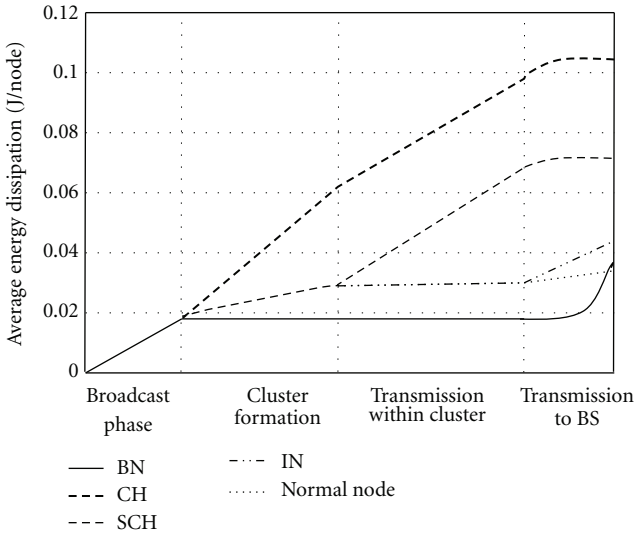



Figure 6: Average dissipated energy of each category in a round time.

average energy dissipation of BNs is decreased about 30%–50% through utilizing BN sleep strategy.

In Figure 7(b) we compared the average energy consumption of all sensor nodes in two mechanisms, respectively, with flexible MAC protocol and without. Flexible MAC protocol means the power control strategy we designed for respective phase. In phases of cluster formation and data transmission within clusters, BNs turn off radio and keep in sleep mode; in phase of data transmission from CHs to BS, only INs/BNs keep awake in their active time. The energy conservation amount by using this strategy reaches 25%–45%. It is noteworthy that, with the increase of percentage of CHs, the amount of energy saving declines. It is due to the number of paths of multihop data transmission from CHs to BS increases, correspondingly the number of joint INs increases.

In order to show the superiority of the performance on energy saving, we compared our proposed protocol with the clustering based protocols, LEACH and HEED. It should be noticed that the routing protocols of data forwarding from CHs to BSs are different in each work. In LEACH, the authors assume the CHs can communicate with the BS directly after data collection and aggregation, while in HEED, the authors utilized the TinyOS beaconing approach, which constructs a breadth-first spanning tree rooted at the BS. It is obvious that the former does not fit for the large-scale WSNs for sensor nodes cannot communicate with the BS directly when the distance increases, whereas the latter does not consider for the reliability and fault tolerance of data transmission. The strategies for improving reliability of data transmission also cause extra energy consumption because they require extensive message exchange. But by adopting more optimal CH selection algorithm and flexible MAC protocol, we also effectively decrease the energy consumption. We compared the network lifetime for three protocols when the number of nodes is 500 and 1000, respectively. The results are shown in Figure 8. The round number for the first and last node dies in each protocol are illustrated, which indicates the network lifetime. From Figure 8 we can observe that our proposed protocol extends lifetime above 140% than LEACH and 50% than HEED. It is because LEACH randomly selects cluster heads and HEED selects cluster heads based on the residual energy of nodes, whereas our proposed algorithm considers the residual energy, number of neighbors (density), and the depth to BS (distance to BS), so that the final cluster heads are well distributed across the network. The flexible MAC protocol also improves the capability of energy saving by turning off node radio when they are idle in the network, since it is well known that idle listening almost causes half of power wastes in wireless sensor networks.

*4.3. Data Delivery Reliability.* In this section we analyzed the reliability of data transmission when using different reliable strategies. As illustrated in Figure 9, with the increasing of node failure rate in WSNs, the average packet delivery ratio decreases correspondingly. Here, the reliability is represented by the probability that a message is successfully delivered, which is calculated as the total number of effective information received at the BS over the total number of effective information initiated from all the sensing nodes. Since we used data aggregation approach, the number of delivered messages cannot be used as the measure parameter. We prefer to use the number of effective information as the measure parameter of data reliability. The data delivery consists of two steps: delivery within cluster and delivery to the BS. The SCH strategy is designed responsible for the failure of data delivery within cluster while multi path routing strategy is designed for the fault tolerance during data delivery from CH to BS.

The faults in WSNs include two types of impact: the node failure and the link failure. In our simulation, we only considered the situation of node failure. It is because that the SCH mechanism is our original and creative strategy, which should be evaluated with high priority. The node failure is random, which includes the situation of either the normal nodes or the CHs (SCHs) fail. The situation of both CH and SCH in the same cluster that have failure simultaneously is also possible in this case. So this fault model is suitable for our research. In previous works [6, 7, 12, 14], the impact of multipath routing protocols has been fully discussed.

As shown in Figure 9, in the mechanism using both SCH and multipath routing mechanisms, the decrease of data delivery ratio is not significant; whereas the mechanism without SCH or multipath routing algorithm decreases very severely. The mechanism only with multipath discovery algorithm performs between the formers. It proved that both SCH approach and multipath routing algorithm are effective in improving the reliability of data delivery.

## 5. Conclusion

In the paper, we proposed a cluster-based energy-efficient and reliable mechanism for WSNs, which integrates the advantages of clustering hierarchical structure and fault tolerant systems. Based on the functionalities in WSNs, sensor nodes are classified into five categories. Each category

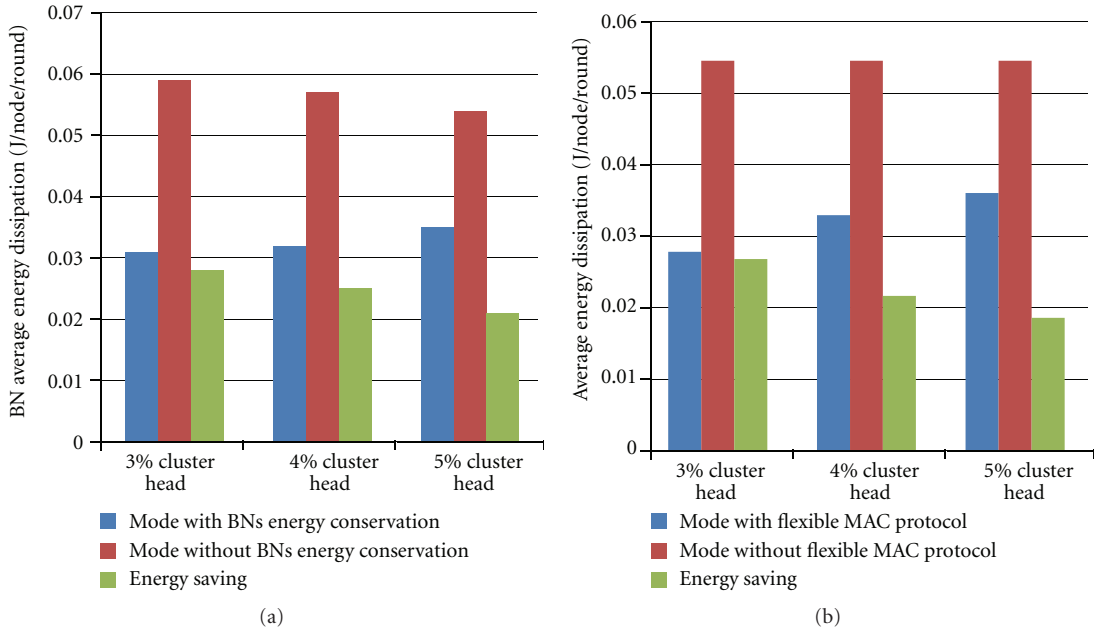



Figure 7: (a) Comparison of average energy dissipation of BNs with/without BN energy conservation strategy. (b) Comparison of node average energy dissipation with/without flexible MAC protocol.

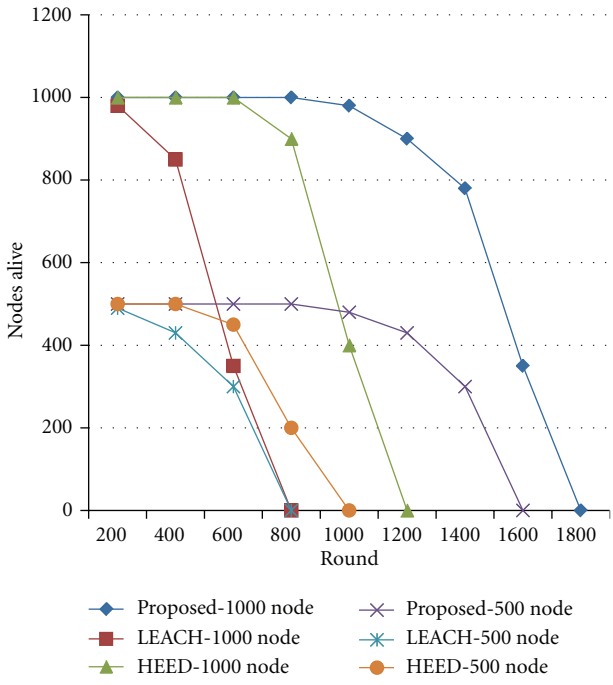



Figure 8: Comparison of network lifetime.

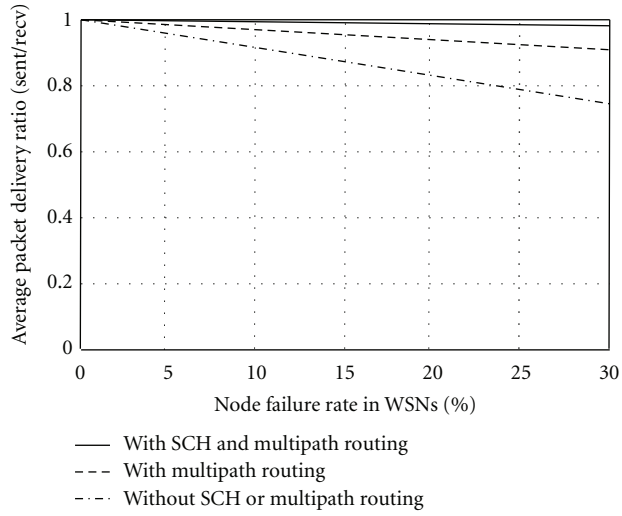



Figure 9: Comparison of average packet delivery ratio.

executes specific network and MAC protocol and processing operations. Using this flexible strategy, our proposed mechanism provides maximal reliability, energy efficiency, and scalability.

It is worth mentioning that some strategies for reliability and fault tolerance cause extra energy consumption whereas strategies for energy efficiency also impact the performance of reliability. For example, the flexible MAC protocol decreases energy consumption greatly, meanwhile it is possible that nodes cannot wake up timely to participate network due to the power control algorithm. The tradeoff between the energy efficiency and reliability is a critical issue in the future. As future works the proposed mechanism will be evaluated in a real environment.

## References

[1] M. Al Ameen, S. M. R. Islam, and K. Kwak, "Energy saving mechanisms for MAC protocols in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2010, Article ID 163413, 16 pages, 2010.

[2] Shio Kumar Singh, M. P. Singh, and D. K. Singh, "A survey of energy-efficient hierarchical cluster-based routing in wireless sensor networks," *International Journal of Advanced Networking and Applications*, vol. 2, no. 2, pp. 570–580, 2010.

[3] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*, vol. 30, no. 14-15, pp. 2826–2841, 2007.

[4] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

[5] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

[6] Y. Challal, A. Ouadjaout, N. Lasla, M. Bagaa, and A. Hadjidj, "Secure and efficient disjoint multipath construction for fault tolerant routing in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1380–1397, 2011.

[7] W. Lou and Y. Kwon, "H-SPREAD: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 4, pp. 1320–1330, 2006.

[8] Y. Jing, X. Mai, X. Jinfu, X. Baoguo, and H. Lu, "A cluster-based multipath delivery scheme for wireless sensor networks," in *Proceedings of the 2nd IEEE International Conference on Broadband Network and Multimedia Technology (IEEE ICBNMT'09)*, pp. 286–291, Beijing, China, October 2009.

[9] S. M. Xiong, L. M. Wang, and J. Y. Wu, "Energy-efficient hierarchical topology control in wireless sensor networks using time slots," in *Proceedings of the 7th International Conference on Machine Learning and Cybernetics (ICMLC'08)*, pp. 33–39, chn, July 2008.

[10] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'02)*, vol. 3, pp. 1567–1576, 2002.

[11] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 293–315, 2003.

[12] H. Huang, Y. Xu, Y. E. Sun, and L. Huang, "Cluster-based load balancing multi-path routing protocol in wireless sensor networks," in *Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA'08)*, pp. 6686–6691, Chongqing, China, June 2008.

[13] N. Sun, Y. B. Cho, and S. H. Lee, "A distributed energy efficient and reliable routing protocol for wireless sensor networks," in *Proceedings of the 14th IEEE International Conference on Computational Science and Engineering*, pp. 273–278, Liaoning, China, August 2011.

[14] A. Boukerche, R. W. N. Pazzi, and R. B. Araujo, "Fault-tolerant wireless sensor network routing protocols for the supervision of context-aware physical environments," *Journal of Parallel and Distributed Computing*, vol. 66, no. 4, pp. 586–599, 2006.

[15] G. Gupta and M. Younis, "Fault-tolerant clustering of wireless sensor networks," in *Proceedings of the IEEE Conference on Wireless Communications and Networking*, vol. 3, pp. 1579–1584, New Orleans, La, USA, March 2003.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 485345, 9 pages
doi:10.1155/2012/485345

*Research Article*

# Optimal Adaptive Antijamming in Wireless Sensor Networks

**Yanmin Zhu,[1,2] Xiangpeng Li,[1] and Bo Li[1,3]**

[1] *Department of Computer Science and Engineering, Shanghai Jiao Tong University, Minhang, Shanghai 200240, China*
[2] *Shanghai Key Lab of Scalable Computing and Systems, Shanghai Jiao Tong University, Minhang, Shanghai 200240, China*
[3] *Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong*

Correspondence should be addressed to Yanmin Zhu, yzhu@cs.sjtu.edu.cn

Received 19 July 2012; Revised 12 October 2012; Accepted 8 November 2012

Academic Editor: Regina B. Araujo



Wireless sensor networks have been widely applied to various application domains such as environmental monitoring and surveillance. Because of reliance on the open transmission media, a sensor network may suffer from radio jamming attacks, which are easy to launch but difficult to defend. Attacked by jamming signals, a sensor network may experience corrupted packets and low network throughput. A number of defense techniques have been proposed. However, each defense technique is suitable for only a limited range of network and jamming conditions. This paper proposes an adaptive approach to antijamming for sensor networks by combining the strength of state-of-the-art antijamming techniques, which enables each node to adaptively select the optimal antijamming technique for different jamming conditions. The great challenge is that the sensor network undergoes varying jamming conditions over time. We address this challenge by formulating the antijamming problem of the sensor network as a Markov decision process and propose an efficient algorithm for computing the best antijamming strategy. By comprehensive simulation experiments, we demonstrate that a sensor network using the derived antijamming strategy can well defend from radio jamming attacks and in the meanwhile retain high energy efficiency.

## 1. Introduction

With the appealing characteristics of low-cost, easy to deploy, and unattended operation and the ability of withstanding harsh environmental conditions, wireless sensor networks have been implemented in a wide range of applications, such as environment monitoring [1] and event detection [2].

Sensor networks transmit wireless signals over the open shared media. This leaves a sensor network vulnerable to radio jamming attacks. In [3, 4], several jamming attacks have been explored, which corrupt control packets, such as RTS (Request-to-Send) and CTS (Clear-to-Send). The jammer just keeps sending packets like RTS to prevent transmission of legitimate packets. These methods are usually based on the statistics of packet transmission history and can cause severe damage to the sensor network with only modest overhead.

Thus, antijamming is enormously important for secure operation of sensor networks. As being well known, sensor nodes are typically powered by batteries and hence limited in power supply. This has been generally accepted as one of the crucial issues of the sensor network. Therefore, antijamming needs to be energy efficient.

A number of antijamming methods have been proposed, such as channel surfing [5], error correction codes and transmission power adjustment. However, these existing countermeasures of jamming attacks are usually suitable for a limited range of jamming conditions with varying operation cost. In the real world scenario, jamming attacks may be very different in nature and may change over time. In addition, radio signals are unstable as many factors may cause jamming signal attenuated in different ways for different environments. As a result, different nodes suffer different degrees of radio jamming. Thus, it is inefficient for a whole sensor network simply to apply a single antijamming technique. This may result in poor performance of antijamming and/or still suffer serious performance degradation of energy consumption.

As shown in Figure 1, sensor nodes in the jamming area may experience different degrees of jamming attack.

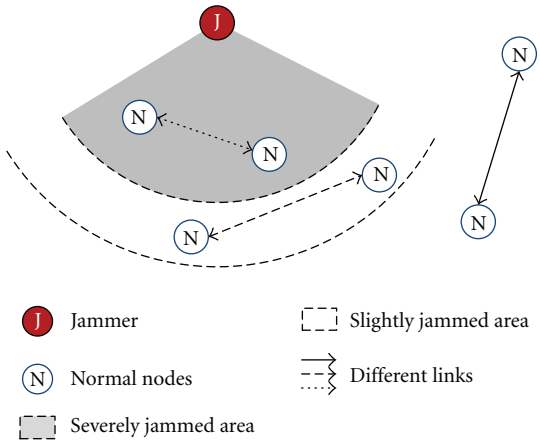



Figure 1: Illustration of varying jamming effects on different nodes.

Thus, the quality of different links varies. For ease of understanding, we show only two levels of jamming signal strength and result in two regions: severely jammed and slightly jammed. In the severely jammed area, the nodes must adopt a heavy antijamming technique in order to transmit a packet successfully. Such a technique usually requires a high energy consumption rate. In the slightly jammed area, the link may experience lower delivery ratio but is still able to deliver data. In this case, a light antijamming technique may be employed for improving the link quality.

This paper focuses on proposing an adaptive approach to antijamming for sensor networks, by combining the strength of different antijamming techniques. With this solution, a sensor node is able to select the best antijamming techniques for different jamming conditions. The challenge is that the sensor network may undergo different jamming conditions over time. As a result, an antijamming technique that is good for the present may produce poor performance after the network condition changes. Importantly, the overhead for a sensor node switching from one antijamming technique to another is usually nonnegligible. In this paper, we solve this challenging issue by formulating the antijamming problem of the sensor network as a Markov decision process and propose an algorithm for computing the best antijamming strategy. Conclusive performance results demonstrate that our approach with the derived antijamming strategy allows a sensor network to perform well against radio jamming attacks.

The rest of this paper is organized as follows. In Section 2, we present the related work. The system model and problem formulation are described in Section 3. In Section 4, we elaborate the design of our MDP-based algorithm and the determination of the optimal antijamming strategy. We present some discussion issues on the design in Section 5. The evaluation results are discussed in Section 6. We finally conclude the paper in Section 7.

## 2. Related Work

Since radio jamming attacks have been recognized as a crucial issue in sensor networks, a plenty of research has been conducted. A good survey for radio jamming attacks and counter measures against radio jamming in sensor networks can be found in [6].

*2.1. Jamming Attacks.* In the literature, many possible jamming attacks have been studied or even implemented in the context of wireless sensor networks.

In [7], it is pointed out that jamming attacks can effectively cause a denial of service of either transmission or reception functionalities. These attacks can easily be accomplished by emitting a radio signal on a particular channel. Different jamming attacks may be posed against a sensor network.

In [8], the authors introduce four different jamming attack models. They explore various detecting techniques for jamming attacks in sensor networks. The packet delivery ratio (PDR) is used to classify a poor radio link, and then a consistency check is performed to make sure whether it is caused by jamming or not. This method is very efficient in identifying the adversary. Nevertheless, it does not provide effective countermeasures against jamming.

In [3], it shows that encrypting packets help to prevent jammers from launching attacks based on the content of the packets. However, temporal intervals of packets induced by the nature of the protocol may release patterns. Such patterns can be exploited by jammers even when packets are encrypted. The authors study packet interarrival times of three representative MAC protocols, S-MAC, LMAC, and B-MAC. And they develop several jamming attacks that allow a jammer to compromise S-MAC, LMAC, and B-MAC in a energy-efficient fashion.

In [9], the authors investigate a scenario where a radio jammer is strategic, meaning that the jammer controls the probability of jamming and the transmission range to maximize the damage to the sensor network in terms of the number of corrupted links. The jammer stops jamming attacks when it is detected by a monitoring node.

In [4], the authors analyze the effect of radio jamming against a sensor network. When the jamming is on the physical layer, the analysis shows that the loss is proportional to the jammer power. On the other hand, the jamming can be launched on the link layer or able to exploit the semantics of MAC-layer packet transmissions. For example, when CSMA/CA is used as the MAC protocol, the jammer could detect transmissions of RTS and CTS frames. Thus, the jammer can selectively jam those control frames.

In [10], reactive jamming is studied, by which a jammer only targets to jam packets already on the air. When jamming only selected packets, the risk of the jammer being detected by the sensor network is minimized. Previously, it was thought that reactive jamming is too challenging for an attacker to implement. The authors demonstrate that flexible and reliable software-defined reactive jamming is practical and easy to implement.

*2.2. Antijamming Techniques.* Due to the serious threats that may be caused by radio jamming attacks, a number of antijamming techniques have been proposed.

In [11], the paper focuses on the performance analysis of various error control codes in terms of BER performance and power consumption. The authors implement different error control codes using VHDL on FPGA and ASIC. In addition, the energy consumption for different error control codes is also measured. BER is the performance metric, evaluated by transmitting randomly generated data through a Gaussian channel. They found that binary-BCH codes with ASIC implementation are best suitable for wireless sensor networks. In the presence of jamming attacks, the channel condition becomes worse, and error control codes can help reduce BER.

One possible solution to cope with radio jamming is to adapt the transmission power of nodes with respect to the power of the jamming radio [12]. It is found that the effect of jamming upon source-receiver communications is not isotropic. The effect of jamming is studied by improving the transmission power on a testbed with Mica2 motes.

In [9], the author formulates the jamming issue as an optimization problem. By solving the optimization problem at both the network and jammers, they can control their probability to transmit the radio signals, so as to achieve the optimal jamming and defense effectiveness. This work studied the interaction between jammer and the nodes in the networks.

In [13], the author proposes a fast jamming detection algorithm in wireless sensor networks. It collectively evaluates the PDR in a given area instead of a pair of nodes, which allows the node detect jamming in a much faster way. In [14], the paper talked about the insider jamming. An attacker compromises some legitimate nodes to acquire cryptographic information and then jams the network. The solution of this paper is to determine the channels by the group key shared by all nodes. When insider jamming happens, the network will generate a new group key for the noncompromised nodes to protect the network from the insider jamming.

In [14], jamming attacks from insiders are studied. An attacker may gain the common cryptographic information through those compromised nodes and then launch jamming attacks. The paper then proposes a compromise-resilient antijamming scheme to deal with the insider jamming problem. According to the scheme, the physical channel used by a sensor network is determined by the group key shared by the sensor nodes.

A technique called channel surfing [15, 16] is developed to cope with the jamming interference, by which the sensor nodes change the communication channel when they detect jamming attacks. Two channel surfing methods are explored. One is coordinated channel switching, in which the entire sensor network changes the radio channel. The other is spectral multiplexing, in which the nodes in the jammed area change the radio channel and the nodes on the boundary act as relays.

In [17], a jammed area detecting and mapping service is developed. As a result, this service allows network applications to reason about the region as an entity. Evaluation results show that regions can be mapped in 1–5 seconds.

*2.3. Summary.* This paper complements the existing work of antijamming by combining the strength of different antijamming techniques and proposes an adaptive antijamming solution to wireless sensor networks.

## 3. Model and Problem Formulation

In this section, we first present the performance description and then introduce the antijamming techniques considered in this paper. Note that it is possible to include more antijamming techniques and the proposed algorithm is still valid.

*3.1. Problem Description.* In a wireless sensor network, there are a set of nodes, denoted as N and a set of a few jammers in the environment, denoted as J. For each node, there are K antijamming techniques available for different jamming conditions.

Since the jamming signal is not constant but varying over time, the node periodically evaluates the channel condition, $\varphi_n(t)$. The period is denoted as $\tau$. Based on $\varphi_n(t)$, the node chooses the proper antijamming technique to deal with different jamming signal. For the node, each antijamming technique has different cost, $C_k$. Focusing on the energy efficient purpose, we evaluate this cost as a function of the energy that the node will consume in the next period. Consider

$$C_{k(t)} = f\left(E_{k(t)}(\tau)\right). \tag{1}$$

In the following period, the performance reward of the node using specific antitechnique is the improvement of the communication between the nodes. Consider

$$R_{k(t)} = (Q(t)) = Q(t+\tau) - Q(t), \tag{2}$$

where $R_{k(t)}$ is the performance reward, and $Q(t)$ is the communication quality.

The objective of the nodes is to choose the proper antijamming techniques, which considers both the reward and the cost. From the definition earlier, we know that $C_{k(t)}$ and $R_{k(t)}(Q(t))$ are different kinds of quantity. Thus, we make the objective by minimizing the product of cost and reward. We focus on a relatively long time period, denoted as $T$, so the nodes totally make $\lfloor T/\tau \rfloor$ decisions. Then, we formulate the overall objective as:

$$\min_{k(t_i)\in\Delta} \quad \sum_{i=0}^{T/\tau} R_{k(t_i)}(Q(t))C_{k(t_i)}. \tag{3}$$

*3.2. Antijamming Techniques.* There are a number of different antijamming techniques in the literature. Without loss of generality, this paper assumes that each sensor node is capable of applying three representative antijamming techniques.

*3.2.1. Transmission Power Adjustment.* With this technique, a sender node increases its transmission power, and thus

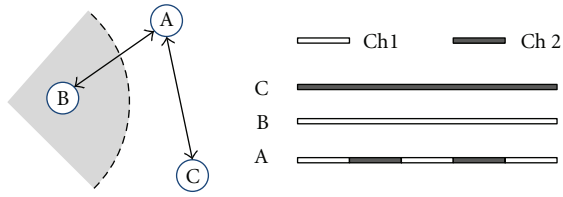



Figure 2: Illustration of the channel hopping technique.

increases the SNR at the receiver node [12]. This technique is suitable under a slight jamming condition, for example, at the periphery of the jamming area. In that area, the jamming signal is relatively weak, so the nodes usually only need to raise its transmission power by one or two levels. This technique introduces modest energy cost.

*3.2.2. Error-Correcting Code.* An error-correcting code [11] is used for correcting some error bits that occurred during transmission. Before transmission, the node encodes the packet. When the receiver has received the packet, the decoding process is capable of correcting some error bits by using the redundancy information contained in the encoded packet (under a certain condition, e.g., the number of error bits is smaller than a given threshold).

Applying error-correcting codes as an antijamming technique is energy efficient as it largely relies on computation and transmission of extra bits. Many error-correcting codes have been proposed. This paper chooses Reed-Soloman code as an example. Other codes can similarly work with our algorithm.

*3.2.3. Channel Hopping.* With this technique [16], a sensor node will change the working channel when it detects strong jamming signals in the current channel. As shown in Figure 2, node B in the shaded area is jammed. Node A is an intermediate node which works on two channels. It switches between the two channels, so it can keep the network connected. When it changes its working channel, it will notify its neighbor working on the same frequency immediately. The schedule of the intermediate nodes is shown in the right of Figure 2. When nodes B and C transmit periodically, and the schedule of Node A is appropriate, there will be no packet loss.

Thus, each sensor node has a set of available antijamming techniques, denoted as $\Delta = \{\delta_0, \delta_1, \delta_2, \delta_3\}$, representing null technique, transmission power adjustment, error-correcting code, and channel hopping, respectively.

## 4. Optimal Adaptive Antijamming

In this section, we present our algorithm in detail, which formulates the problem as a Markov decision process and obtains an optimal policy for antijamming.

We formulate the problem as a 4-tuple: $(S, \Delta, P, C)$, where $S$ is the set of all the node states that describe the channel conditions, $\Delta$ is the set of antijamming techniques, $P_\delta(s, s')$ is the probability that the node state becomes $s'$ after technique $\delta$ is performed at state $s$, and $C$ is the cost of the antijamming technique. In the following, we introduce these four components in detail.

**input:**
PDR: Packet Delivery Ratio
RSSI: Received Signal Strength Indicator
CAS: Current Antijamming Strategy
$\Phi$: The threshold of PDR for normal communication
$K$: The threshold of RSSI for weak link
**output:**
$S$: System State
**main procedure:**
**if** PDR $\geq \Phi$
**return** $S = \min(\Pi(\text{CAS}, \text{PDR}), \Gamma(\text{RSSI}))$
**else if** RSSI $< K$
**return** NormalState
**else**
**return** $S = \max(\Pi(\text{CAS}, \text{PDR}), \Gamma(\text{RSSI}))$
**end if**

Algorithm 1: System state determination.

*4.1. System States.* The system states denote the channel conditions $\varphi_n(t)$, which describe the different degrees of the jamming conditions.

In [8], the authors use PDR and RSSI (Received Signal Strength Indicator) to denote jamming signals with good accuracy. In our algorithm, however, the antijamming strategy affects the value of PDR. It will not be so accurate for using those two parameters to describe the current channel condition. We will use a method that considers the current antijamming strategy as well.

Considering the limited computing ability of the sensor nodes, it is important to reduce the complexity of the algorithm. Therefore, we use only four states, denoted as $S = \{0, 1, 2, 3\}$, which represent four different jamming conditions and correspond to the three antijamming strategies plus the case requiring no countermeasures. For the same reason, we use five levels for RSSI. As Algorithm 1 shows, functions $\Pi(\text{CAS}, \text{PDR})$ and $\Gamma(\text{RSSI})$ output the system state depending on the current countermeasure. For each countermeasure, it is suitable for a certain level of jamming, so there is a correspondence between states and countermeasures. Thus, when the PDR value is high enough, it indicates that the current antijamming strategy is effective. Then, function $\Pi(\text{CAS}, \text{PDR})$ outputs the corresponding state. For the transmitting power of the node is always the same, the RSSI value will be in a certain level of the normal case. When the jammer is present, the RSSI value will also rise. Under different jamming levels, the RSSI value will be different. Making this correspondence between RSSI and the jamming conditions, we realize function $\Gamma(\text{RSSI})$. There is a special interval of RSSI value, which is below the normal signal strength, meaning that the link is weak. In that case, the algorithm will return a normal state, because a low PDR value is not caused by radio jamming.

The smaller the value of the state is, the better the link condition is. When the PDR value is high, we prefer a light antijamming strategy. Thus, we choose the smaller state. When the PDR is not that good, for the sake of effectiveness, we choose a worse case as the system state.

*4.2. Transition Probability.* Since the state of nodes changes due to the varying jamming conditions, the transition probability of the states also describes the variation of jamming signals.

The transition probability is acquired by analyzing historical data. The nodes record number $k_i^\delta$ of the node reaching the state $i \in S$ and performing action $\delta$. If the state changes to $j$ at the current period, then the nodes add one to the variable $k_{i\to j}^\delta$ that keeps the total number of the state transitions from $i$ to $j$ for action $\delta$. When $S(t-\tau) = i$ and $S(t) = j$, then the transition probability can be calculated as

$$\begin{aligned} P_\delta(S(t-\tau), S(t)) &= \frac{\Pr(S(t-\tau), S(t), \delta)}{\Pr(S(t-\tau), \delta)} \\ &= \frac{k_{i\to j}^\delta / k_i}{k_i^\delta / k_i} = \frac{k_{i\to j}^\delta}{k_i^\delta}, \end{aligned} \tag{4}$$

where $k_i$ is the total number of nodes that reach the state $i$.

*4.3. Cost of Antijamming Techniques.* The cost function is about the energy consumption caused by the antijamming techniques.

*4.3.1. Adjusting Transmission Power.* The cost of increasing transmission power is easy to compute. It is the raised power multiplied by the packet transmission time,

$$C_I = \sum_{i=1}^{n} P_{\text{tx}} t_{\text{pkt}} = \sum_{i=1}^{n} \frac{P_{\text{tx}} L_{\text{pkt}}}{R_{\text{bits}}}, \tag{5}$$

where $C_I$ is the cost of the increasing power action; the time of transmitting every packet is $t_{\text{pkt}}$; $L_{\text{pkt}}$ and $R_{\text{bits}}$ represent packet length and the bits transmission rate, respectively; $n$ is the total number of packets within one period $\tau$ when the node is performing that technique.

*4.3.2. Error-Correcting Codes.* The energy consumed by this technique is the power used for the encoding and decoding process and transmitting the redundant bits. For the error-correcting codes has to be undertaken by both of the communicating nodes, the notification process also consumes extra energy.

As the notification process is all the same and will not change, therefore, the energy expended is a constant. With this information, we have the cost function of the error-correcting codes technique as

$$C_{\text{EC}} = \sum_{i=1}^{n} \left( \frac{P_{\text{tx}} L_{\text{EC}}}{R_{\text{bits}} + E_{\text{dec}}} \right) + E_{\text{noti}}. \tag{6}$$

$L_{\text{EC}}$ is the length of the encoded packets. Since there are more bits than the normal packets, $L_{\text{EC}}$ will be bigger than $L_{\text{pkt}}$. For this method, the nodes transmit the signals in a normal level of power. The first component of (6) is the energy for transmitting the packets. $E_{\text{dec}}$ is the energy spent for decoding the packets or correcting the errors of the received signal. In [18] the author gives a method to calculate the computing energy cost.

$E_{\text{noti}}$ is the energy expended for the notification process. Because this process is invariable, it just equals the multiplication of the normal power level and the time spent for this process.

Added up this tree part of the energy, we will get the total energy consumed by the error-correcting strategy.

*4.3.3. Channel Surfing.* The nodes also transmit their signals in normal power in this technique. As the error-correcting code strategy, the nodes have to notify the neighbors that it will change to another channel, for the neighbors undertaking the same strategy to keep the connectivity of the network. Secondly, when the nodes take this strategy, the intermediate nodes also need to send some packets when it switches to each channel. The total cost could be

$$C_{\text{CS}} = \sum_{i=1}^{m} \frac{P_{\text{tx}} L_{\text{noti}}}{R_{\text{bits}}} + \sum_{i=1}^{n} \frac{P_{\text{tx}} L_{\text{pkt}}}{R_{\text{bits}}} + E_{\text{noti}}. \tag{7}$$

$E_{\text{noti}}$ is just like the error-correcting codes, for the notification process is all the same. In this strategy, the node has to inform the neighbors when it goes to a new channel. The $P_{\text{tx}} L_{\text{noti}} / R_{\text{bits}}$ is the energy expended to send those packets. $m$ means the total channel switches. The energy consumed by data packets is the second part in (7).

*4.4. Policy Determination.* In the following, we use PDR to describe performance reward,

$$R_{\delta(t-\tau)}(Q(t-\tau)) = \text{PDR}(t) - \text{PDR}(t-\tau), \tag{8}$$

and we define:

$$\gamma_{i\delta} = 1 - \overline{R}_{\delta(t)}(S(t)), \tag{9}$$

$$\lambda_{i\delta} = \gamma_{i\delta} C_\delta, \tag{10}$$

where $\lambda_{i\delta}$ is the cost of technique $\delta$ at state $i$.

Because $\gamma_{i\delta}$ is a coefficient of the cost, (9) makes the more effective technique that has higher reward and less energy cost. It results in a smaller $\gamma_{i\delta}$ when the jamming is severe. Then, the cost $\lambda_{i\delta}$ will become less than the techniques with less energy cost, for the value $\gamma_{i\delta}$ of the lighter technique is probably greater than one. Therefore, the node is more likely to perform a more effective technique to guarantee a certain communication quality. On the other hand, when the jamming is not serious, a heavy technique does not gain so much reward that makes the cost less than the lighter ones. In such a case, the technique with less energy cost is preferred.

Based on the previous definitions, we devise the policy improvement algorithm to solve the MDP problem. The details of this algorithm are explained as follows.

We denote the total expected cost of the node beginning in state $i$ and evolving for $n$ periods by $\varepsilon_i^n(D)$, where $D$ is the related policy. Then, we have

$$\varepsilon_i^n(D) = \lambda_{i\delta} + \sum_{j=1}^{M} P_\delta(i,j)\varepsilon_j^{n-1}(D), \quad \text{for } i = 1,2,\ldots,M, \tag{11}$$

where $\lambda_{i\delta}$ is the cost introduced in the first period. The second part of the equation is the cost of the next $n$ period. There are $M$ states in total. The long run expected average cost per unit time could be expressed as

$$\zeta(D) = \sum_{j=1}^{M} \pi_i \lambda_{i\delta}, \tag{12}$$

where $\pi_i$ is the steady distribution of the states. We can have an approximate relationship when $n$ is large as follows:

$$\varepsilon_i^n(D) \approx n\zeta(D) + \varepsilon_i(D). \tag{13}$$

We can consider $\varepsilon_i(D)$ as the effect on the total expected cost due to beginning in state $i$. After substituting (13) into (11), we get

$$\zeta(D) = \lambda_{i\delta} + \sum_{j=1}^{M} P_\delta(i,j)\varepsilon_j(D) - \varepsilon_i(D), \quad i = 1,2,\ldots,M. \tag{14}$$

The policy improvement algorithm starts by choosing an arbitrary policy $D_n$ and set $\varepsilon_M(D_n) = 0$. Then, it solves (14) to $\zeta(D_n), \varepsilon_1(D_n), \varepsilon_2(D_n), \ldots, \varepsilon_M(D_n)$. We use $\varepsilon_i(D_n)$ to find another policy $D_{n+1}$ such that for each state $i$,

$$\min_{\delta\in\Delta} \quad \lambda_{i\delta} + \sum_{j=1}^{M} P_{i,j}(\delta)\varepsilon_j(D) - \varepsilon_i(D), \tag{15}$$

where $\delta = d_i(D_{n+1})$. When $D_{n+1}$ and $D_n$ are identical, this iteration process will stop. Otherwise, it sets $n = n + 1$ and this process continues.

*4.5. Algorithm Framework.* The framework of the optimal antijamming algorithm is shown in Algorithm 2.

Considering the jamming pattern may change over time, each policy has an effective period, as shown in Algorithm 2. When the effective period runs up, the nodes will determine a new policy to make sure that the current policy is suitable for the jamming pattern. After the policy is acquired, the nodes begin to communicate with each other. The communication period allows a node to obtain PDR. The node then determines the state using the algorithm introduced before. Then, it updates the transition probabilities, which is for later policy determination. Before the next period of communication, the nodes choose a proper antijamming strategy based on the current policy, and it will send a notification to make sure the nodes are using the same antijamming strategy.

```
while (1) do
    while (PolicyEndureTime) do
        while (CommunicationPeriod) do
            Nodes Communicate;
        end while
        GetPDR();
        DetermineNodeState();
        UpdateTransitionProbabilities()
        ChooseStrategy(Policy, State);
        SendNotification();
    end while
    DeterminePolicy();
end
```

Algorithm 2: Antijamming algorithm framework.

## 5. Discussions

In this section, we discuss two important design issues when designing the adaptive antijamming techniques for wireless sensor networks.

*5.1. Integration of Multiple Antijamming Techniques.* It is important for individual sensor nodes in a sensor network to work collaboratively as a whole. Since multiple antijamming techniques are equipped, sensor nodes in the network may be using different antijamming techniques. This may cause network disconnections to the sensor network, resulting in failed packet delivery. For example, when the technique of channel surfing is adopted, two neighboring nodes may be using different channels and make the link between the two nodes broken.

Thus, it is important to develop a coordination protocol which ensures that the network connectivity is maintained even the sensor nodes are using different antijamming techniques. The design of such a protocol is beyond the scope of the paper and is subject to future research.

*5.2. Exploiting More Antijamming Techniques.* In this paper, we have discussed three typical antijamming techniques. However, as mentioned in the Section of Related Work, there are many more antijamming techniques. Then, it is an important problem on what antijamming techniques should be equipped on sensor nodes. Note that the increasing number of equipped antijamming techniques may consume more resources such as memory and energy. More importantly, it may also increase the design complexity of the coordination protocol mentioned previously. Thus, the selection of the antijamming techniques should balance the advantage brought by more antijamming techniques and the overhead introduced.

## 6. Performance Evaluation

In this section we first present the performance evaluation metrics and the simulation setup and then discuss the evaluation results.

Table 1: Simulation parameters.

| Parameter | Value |
|---|---|
| Region | 40 m by 40 m |
| Propagation model | Large-scale path loss |
| Loss component $n$ | 4 |
| Transmission power | −5 dBm |
| Adjusted power | −3 dBm |
| $I_{tx}$ | 13.9 mA |
| $I_{rx}$ | 19.7 mA |
| Voltage | 3 v |
| Data packet length | 50 Bytes |
| Ack packet length | 20 Bytes |
| Transmission rate | 250 Kb/s |

*6.1. Methodology and Simulation Setup.* We have developed a discrete event driven simulator for simulating jamming attacks and antijamming operations of a sensor network. More specifically, we simulate the details of radio propagation of sensor nodes. The main simulations parameters are listed in Table 1.

We adopt the following simulation setting. The sensor nodes are uniformly distributed in a square area of 40 m by 40 m. We use the large-scale path loss model to describe the attenuation of a radio signal. The parameters of energy consumption are compatible with the CC2420 Chip. The nodes in the network transmit signals with the power of −5 dBm, and the jammer randomly chooses the power from (0, −1, −3, −5, −7, −10, −15, and −25) dBm. The raised power is −3 dBm, which is one level higher. The current for transmission is $I_{tx} = 13.9$ mA, and the current for reception is $I_{rx} = 19.7$ mA. The typical voltage is $V = 3$ v. The jammer with higher power means that the nodes near the jammer will suffer more severe damage. The jammer is placed in the middle of the network so that it jams as many nodes as possible. For the large scale path loss model, we set the value of the path loss exponent $n = 4$, and the reference distance $d_0 = 1$. The packet length $L_{ACK}$ is 25 Bytes, and $L_{data}$ is 50 Bytes. These values comply with the IEEE 802.15.4 Standard. They are convenient for the error-correcting codes, which is RS (31, 25, 3). The bit transmission rate $R_{bits}$ is 250 Kb/s.

*6.2. Packet Delivery Ratio.* We first study the packet delivery ratio of different antijamming strategies. Figure 3 shows the effectiveness of the computed antijamming strategies in terms of packet delivery rate. As the normal condition, the nodes that are within 8 meters from the jammer have lost more than 50% packets, which make it difficult for effective communication in sensor networks. The strategy of raising the transmission power can improve the situation, but there are still a lot of nodes which severely suffer from packet loss. As BER affects PDR signifcantly, we can see that the curve of the strategy of using error-correcting Code is much better than the former two strategies. The channel surfing strategy lets the nodes work on another channel which experiences no jamming attack, so the nodes can communicate with each other properly. It is also effective and can maintain the communication quality.

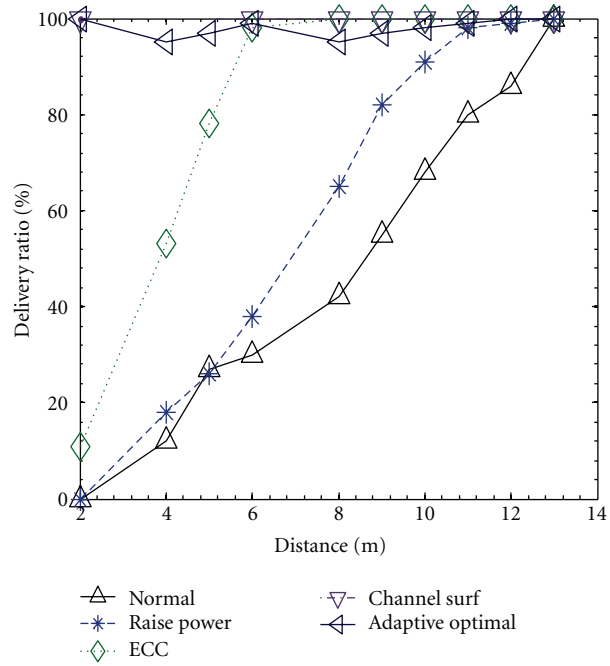



Figure 3: Delivery ratio of different strategies.

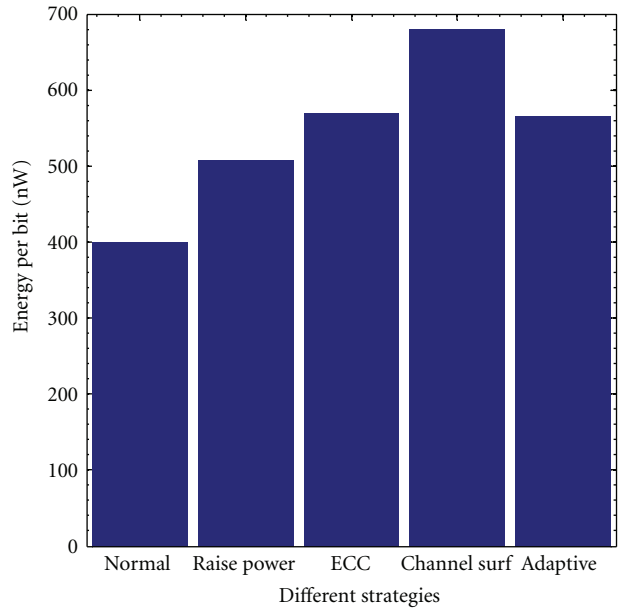



Figure 4: Energy consumption of different strategies.

*6.3. Energy Efficiency.* Next, we investigate the energy efficiency of different strategies. Figure 4 shows the energy consumption of the nodes. We evaluate the energy efficiency of the antijamming strategies by measuring the energy consumed by information bits excluding notification packets and coding redundancy. In the figure, we can find that the computed strategy by our algorithm expends 20% less energy

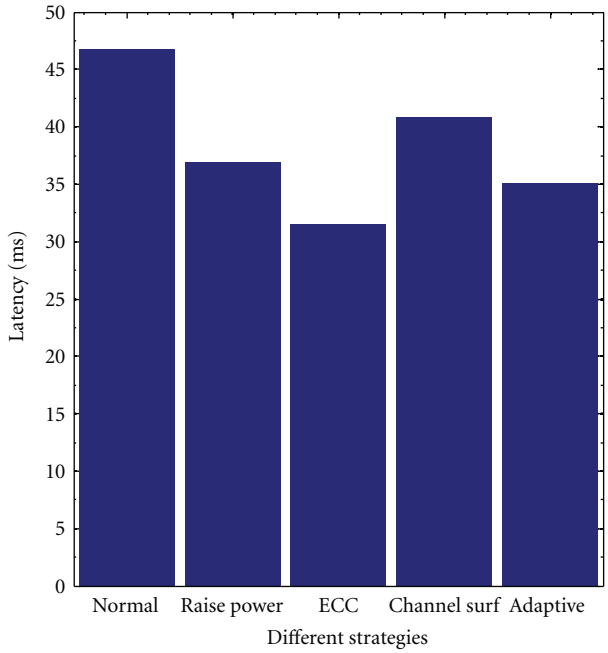



Figure 5: Latency performance of different strategies.

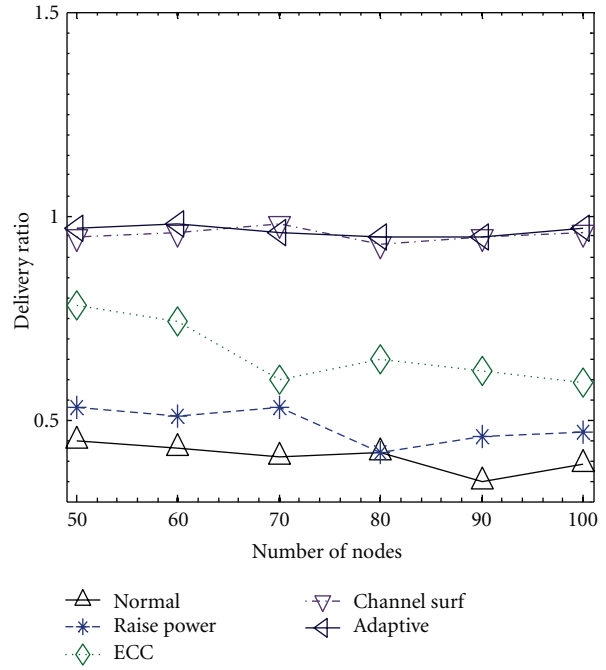



Figure 6: Delivery ratio verses number of nodes.

than the channel surfing strategy which is the most effective strategy.

*6.4. Latency.* We then study the latency performance of different antijamming strategies. The latency of a packet transmission is measured as the time between the instant of time when the node starts contending for the channel and the instant when the acknowledgement is received. In Figure 5, we can see that our adaptive antijamming strategy introduces a latency slightly longer than that of ECC. The channel surfing strategy introduces a much longer latency because it takes a long time for the nodes to change the communication channel. When no antijamming strategy is used in the sensor network, the latency is long because it needed many retransmissions before the packet is successfully delivered.

*6.5. Scalability.* We finally investigate the scalability of different antijamming strategies. In Figure 6, we show the packet delivery ratio when the number of nodes is varied from 50 to 100. We can see that as the number of nodes increases in the network, the delivery ratio of each antijamming scheme slightly drops since there is higher contention for accessing the media. However, we can see that the delivery ratio performance of our adaptive scheme only has a modest drop in packet delivery ratio when there are more sensor nodes in the network. This shows that our adaptive scheme is scalable to the increasing scale of the network.

## 7. Conclusion

We have presented the algorithm for selecting the best antijamming strategy for a sensor network, in which different sensor nodes may experience different degrees of jamming attacks. We propose an approach for combining the strength of several jamming countermeasures and allow a sensor node to adopt the best antijamming technique. Sensor nodes in the sensor network can adaptively change their antijamming methods as the jamming condition changes over time. The comprehensive simulation experiments have demonstrated that our algorithm achieves good performance in terms of successful delivery rate and at the meanwhile consumes slightly more energy.

## Acknowledgments

This research is supported in part by MIIT of China (2009ZX03006-001-01), Shanghai Pu Jiang Talents Program (10PJ1405800), Shanghai Chen Guang Program (10CG11), NSFC (no. 61170238, 60903190, 61027009, 60970106, and 61170237), Doctoral Fund of Ministry of Education of China (20100073120021), 863 Program (2009AA012201 and 2011AA010500), HP IRP (CW267311), SJTU SMC Project (201120), and Program for Changjiang Scholars and Innovative Research Team in Universities of China (IRT1158, PCSIRT).

## References

[1] Y. Liu, Y. He, M. Li et al., "Does wireless sensor network scale? A measurement study on GreenOrbs," in *Proceedings of the IEEE INFOCOM*, pp. 873–881, April 2011.

[2] M. Li, Y. Liu, and L. Chen, "Non-threshold based event detection for 3D environment monitoring in sensor networks," in *Proceedings of the IEEE ICDCS*, 2008.

[3] Y. W. Law, M. Palaniswami, L. V. Hoesel, J. Doumen, P. Hartel, and P. Havinga, "Energy-efficient link-layer jamming

attacks against wireless sensor network MAC protocols," *ACM Transactions on Sensor Networks*, vol. 5, no. 1, pp. 1–38, 2009.

[4] R. Negi and A. Perrig, "Jamming analysis of MAC protocols," Tech. Rep., 2003.

[5] W. Xu, W. Trappe, and Y. Zhang, "Channel surfing: defending wireless sensor networks from interference," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN '07)*, pp. 499–508, April 2007.

[6] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou, "A survey on jamming attacks and countermeasures in WSNs," *IEEE Communications Surveys and Tutorials*, vol. 11, no. 4, pp. 42–56, 2009.

[7] W. Xu, K. Ma, W. Trappe, and Y. Zhang, "Jamming sensor networks: attack and defense strategies," *IEEE Network*, vol. 20, no. 3, pp. 41–47, 2006.

[8] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC '05)*, pp. 46–57, May 2005.

[9] M. Li, I. Koutsopoulos, and R. Poovendran, "Optimal jamming attacks and network defense policies in wireless sensor networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (IEEE INFOCOM '07)*, pp. 1307–1315, May 2007.

[10] M. Wilhelm, I. Martinovic, J. B. Schmitt, and V. Lenders, "Reactive jamming in wireless networks: how realistic is the threat?" in *Proceedings of the ACM WiSec*, 2011.

[11] G. Balakrishnan, Y. Mei, J. Yingtao, and K. Yoohwan, "Performance analysis of error control codes for wireless sensor networks," in *Proceedings of the 4th International Conference on Information Technology-New Generations (ITNG '07)*, pp. 876–879, April 2007.

[12] W. Xu, "On adjusting power to defend wireless networks from jamming," in *Proceedings of the 1st Workshop on the Security and Privacy of Emerging Ubiquitous Communication Systems*, pp. 1–6, 2007.

[13] K. Siddhabathula, "Fast jamming detection in wireless sensor networks," University of Texas, 2011.

[14] X. Jiang, W. Hu, S. Zhu, and G. Cao, "Compromise-resilient anti-jamming for wireless sensor networks," *Information and Communications Security*, vol. 6476, pp. 140–154, 2010.

[15] W. Xu, T. Wood, W. Trappe, and Y. Zhang, "Channel surfing and spatial retreats: defenses against wireless denial of service," in *Proceedings of the ACM Workshop on Wireless Security (WiSe '04)*, pp. 80–89, October 2004.

[16] W. Xu, W. Trappe, and Y. Zhang, "Defending wireless sensor networks from radio interference through channel adaptation," *ACM Transactions on Sensor Networks*, vol. 4, no. 4, article 18, 2008.

[17] A. D. Wood, J. A. Stankovic, and S. H. Son, "JAM: a jammed-area Mapping service for sensor networks," in *Proceedings of the 24th IEEE International Real-Time Systems Symposium (RTSS '03)*, pp. 286–297, December 2003.

[18] P. Lettieri, C. Fragouli, and M. B. Srivastava, "Low power error control for wireless links," in *Proceedings of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '97)*, pp. 139–150, September 1997.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 725421, 19 pages
doi:10.1155/2012/725421

*Research Article*

# LiReTa: A Lightweight Reliable Transmission Scheme for Wireless Sensor Networks Using Cross-Layer Information

**Ga-Won Lee and Eui-Nam Huh**

*Department of Computer Engineering, KyungHee University, Global Campus, 1-Seochon-dong, Giheung-gu, Gyeonggi-do, Yongin-si 446-701, Republic of Korea*

Correspondence should be addressed to Eui-Nam Huh, johnhuh@khu.ac.kr

Received 6 January 2012; Revised 6 May 2012; Accepted 23 May 2012

Academic Editor: Regina B. Araujo



Information delivered through sensor networks is used in industries to increase quality of life (QoL). Lossless data in wireless sensor networks (WSNs) is a communications challenge that stands in the way of accurate data delivery. Although end-to-end data retransmission has evolved as a reliable mode of data transportation for the Internet, it is not applicable to WSNs due to the lack of reliable wireless links and resource constraints in sensor nodes. In this paper, we propose an efficient and reliable overhearing-based data transfer protocol for WSNs by introducing selective "direct acknowledgement ($ACK_{dir}$)" or "implicit acknowledgement ($ACK_{imp}$)" in cross-layer design. This protocol assesses the path (or link) quality and delegates the ACK message if it has good communication on paths. In addition, the protocol uses implicit ACK in order to be energy efficient, reducing traffic. Simulation results show that energy efficiency is improved by 30% compared with other approaches.

## 1. Introduction

In recent years, wireless sensor networks (WSNs) have expanded from simple environmental surveillance and information delivery systems to various mission critical applications such as Ubiquitous (u-) healthcare services, u-agriculture systems, and u-defense. These applications require the reliable delivery of high-priority events to sinks, reliable control and management of the sensor network structure, and the capacity for remote programming/re-tasking of sensor nodes in a controlled, reliable, robust, and scalable manner [1]. Importantly, all of these applications necessitate that all data are to be transmitted without loss within their respective WSNs.

However, unlike traditional networks (e.g., IP networks), reliable data transmission remains a challenge in WSN environments. WSNs are highly distributed self-organized systems that rely on significant numbers of scattered low-cost tiny devices/sensor nodes featuring major limitations with respect to processing, memory, communications, and power capabilities. Since sensor nodes are highly resource constrained, the design of reliable data transmission protocols is very challenging.

Many transport protocols have been proposed and implemented in the literature to improve the reliability of WSNs. These protocols are mainly designed (1) to confirm data transfer and lost data recovery by requesting an acknowledgement (ACK) message (called "direct ACK" denoted as $ACK_{dir}$) or notifying the sender of failure with a negative acknowledgement (NACK) message [1–4], (2) to increase data transfer success rates by delivering via multiple paths [5–7], and (3) to avoid data collision by using event-based approaches or collision detection approaches [8, 9]. To achieve lossless reliable data transfer, these data transmission protocols commonly select feedback message or recovery factors such as ACK, NACK, hop-by-hop recovery, end-to-end recovery, or the number of packet duplications. However, these static quality-based parameters have limitations with error-prone and unstable WSNs, since they do not apply to dynamic conditions in sensor nodes. Moreover, previous approaches focus on minimum data loss without considering energy consumption. Indeed, the lifespan of sensors and the

unreliable nature of WSNs results in two tradeoff problems: energy consumption and lossless data transfer.

Nevertheless, few studies have been devoted to the design of reliable transport protocols using *path reliability* in a multipath routing environment. This approach is primarily used to choose reliable paths based on directed diffusion in cases of delay-sensitive data delivery over error-prone WSNs [10]. Channel error rate is mainly used to measure path reliability; however, this mechanism cannot ensure end-to-end reliability in a WSN environment, which is important for mission critical applications.

In this paper, we propose a Lightweight Reliable data Transmission (LiReTa) method using cross-layer information, which calculates the reliability of *every path (or link) of the node* using the power level received signal strength indicator (RSSI) value and the channel error rate of the node. Moreover, our proposed method overhears communications of neighbor nodes because the source node can overhear the forwarding signal of the receiver as it is sent to the neighbor of the receiver. Thus the sender recognizes successful delivery to its neighbor without receiving the ACK message. This is called "implicit ACK" denoted as $ACK_{imp}$.

The protocol proposed in this paper contributes to a quick error-recovery of pump-slowly fetch-quickly (PSFQ) [3] while it uses generic ACK/NACK for reliable transmission. In addition, energy efficiency is achieved using a selective ACK mechanism that requests ACK selectively if current path reliability is less than a threshold value. Simulations were conducted to show the effectiveness of LiReTa compared with several existing algorithms in terms of energy consumption and traffic reduction.

The rest of this paper is organized as follows: Section 2 introduces some fundamental factors that determine path-reliability and discusses several well-known reliable transfer schemes and their associated problems. Section 3 describes the proposed algorithm. Section 4 shows transmission use case scenarios. Simulation results of the proposed LiReTa algorithm are described in Section 5, and the conclusions of our work are detailed in Section 6.

## 2. Related Work

In this section, we first summarize some issues and problems of sensor networks regarding reliable transmission. We then briefly review some basic approaches for reliable data transfer in wireless sensor networks to show how our research builds on previous work.

*2.1. Reliable Transport Protocols for WSNs.* An exhaustive list and analysis of transport protocols for WSNs can be found in [1, 4, 11]. WSN transport protocols can be classified into five categories: ACK/NACK based schemes, multipath transfer schemes (short multipackets), collision avoidance schemes, and reliability schemes as shown in Table 1 along with some representative methods/examples.

The ACK/NACK mechanism provides *hop-by-hop* or *end-to-end* data dissemination by using ACK and NACK messages in cases of missing sequential packets. This mechanism, when applied to high channel error rate scenarios, causes an ACK implosion problem where the buffer is overflowing due to too frequent retransmission and unnecessary traffic packets and, consequently, is not applicable to WSNs [12].

Table 1: Reliable data transfer methods for WSN.

| Type | Method |
|---|---|
| ACK-based scheme | RM21 |
| NACK-based scheme | PSFQ/GARUDA/RMST |
| Multipackets | HHR/ReInForM/ReTrust |
| Collision avoidance | ESRT/CODA |
| Reliability scheme | Directed diffusion considering reliability |

Whenever a node receives a message, it sends $ACK_{dir}$ messages to notify the sender of transmission success. The ACK implosion problem occurs when multiple ACK messages are simultaneously received. Indeed, ACK implosion causes unnecessary traffic and data loss, which decreases the availability and performance of links due to repetitive message transfers.

Reliable multicast with $ACK_{imp}$ and indirect recovery (RM21) [2] is an ACK-based protocol that uses $ACK_{imp}$ messaging and indirect recovery to reinforce the disadvantages of NACK. When the error rate is low, energy consumption is more efficient and RM21, by utilizing $ACK_{imp}$, consumes less power than methods that send $ACK_{dir}$ messages. However, $ACK_{imp}$ message failure or methods of sensor deployment for RM21 has not been discussed clearly. Furthermore, error recovery rates increase rapidly when error rates increase, thereby reducing energy efficiency.

PSFQ [3] is a protocol that ensures reliability in WSNs. The key idea of the design of PSFQ is to distribute data from a source node by pacing data at a relatively slow speed (pump-slowly), but allowing nodes that experience data loss to fetch quickly (i.e., to recover any missing segments from their local immediate neighbors aggressively) [1]. PSFQ eliminates unnecessary traffic for NACK messages through retransmission requests at a middle node and minimizes the cost of loss recovery by using data localized among immediate neighbors to achieve loose delay bounds. However, the middle node in standby status is unable to transfer lost packets located in the buffer until the next node notices that a packet is missing or retransmission is complete. Thus, the entire data transmission time is much longer and increases the possibility of buffer overflow in the middle nodes.

A scalable approach for reliable downstream data delivery in wireless sensor networks (GARUDA) [1] solves the first sequence packet transfer problems found with NACK protocols. Specifically, it guarantees the reliability of the first packet by using a wait for the first packet pulse (WFP), where the core node acts as a recovery server when the data transmission fails using downstream data. However, energy consumption with this protocol is very high and, consequently, it is inappropriate for WSNs.

Table 2: Reliable data transfer method comparison.

| Protocol | Reliable guarantee | Direction | Energy efficiency | Recovery | Data transfer speed |
|---|---|---|---|---|---|
| RM21 | End-to-end/node-to-node | Up/down | Average | ACK based | Slow |
| PSFQ | End-to-end/node-to-node | Downstream | Low | NACK based | Slow |
| GARUDA | End-to-end | Downstream | Low | NACK based | Initially slow, after average |
| RMST | End-to-end | Upstream | Average | NACK based | Average |
| Heuristic | End-to-end | Upstream | Average | ACK/NACK | Average |
| ReInForM | End-to-end | Upstream | Based on reliability requirement | — | Slow |
| HHR | End-to-end | Upstream | Low | — | Slow |
| ReTrust | End-to-end | Upstream | Based on reliability requirement | — | Slow |
| ESRT | Event reporting | Upstream | Average | — | Average |
| CODA | Event reporting | Upstream | Average | — | Average |
| DD with reliability | End-to-end/node-to-node | Up/down | Low | — | Average |

Reliable multi-segment transport (RMST) [4] is investigated through simulations of the tradeoff in implementation reliability between MAC, transport, and application layers. The conclusion is that hop-by-hop recovery plays an important role in achieving reliability and end-to-end recovery is inadequate. However, packet recovery using source nodes swamps the load to source node in non-caching mode.

In addition to ACK/NACK based protocols, hop-by-hop reliability (HHR) schemes [5] and reliable information forwarding using multiple paths (ReInForM) [6] provide reliable transmission by combining multi-packet and multi-path methods. HHR uses unicast transmission to transfer many copies of a single packet. This scheme considers packet loss rate, packet transfer possibility, and number of hops to create copies of the packet. However, if the channel quality is poor, all of the copies may be wasted.

On the other hand, ReInForM compensates for the disadvantages of HHR by transferring copies through random multiple paths to maintain a specified reliability and to prevent energy inefficiency in good quality paths due to traffic that is dispersed to many nodes. However, the channel error rate increases when the number of hops between source and sink is large, which causes the copies of packets and the number of paths to grow exponentially. Hence, ReTrust [7] improves upon the ReInForM framework by focusing on efficiently reducing such loads using intermediate source/sink (IS) in sensor networks; however, unnecessary traffic delays may remain and load problems may still occur in IS nodes.

In addition to the previous methods used in WSNs as described above, there are other methods such as event-to-sink reliable transport (ESRT) [8], congestion detection and avoidance (CODA) [9], and a method proposed in [10] that applies reliability in the routing path to guarantee path reliability. ESRT employs an event-to-sink reliability model to provide reliable event detection that embeds a congestion control component [3] and manages different events with different levels of reliability. CODA is an energy congestion control scheme that avoids collisions in WSNs and comprises three mechanisms: congestion detection, open-loop hop-by-hop back pressure, and closed-loop; however, such detection of loading states in channels consumes a significant amount of energy.

In [10], a reliable data transfer mechanism using directed diffusion in WSNs is proposed. This mechanism involves selecting a path with higher reach-ability and transferring data along the path chosen. The path choice is based on end-to-end reliability as calculated by the dissemination procedure of the Interest packets, while each node of a sensor network maintains only the information in its neighborhood. [10] only considers channel error rate for routing path reliability. Table 2 presents a comparative analysis of the existing reliable transport protocols for WSNs.

## 3. LiReTa: Lightweight Reliable Data Transmission Method

### 3.1. Preliminaries

*3.1.1. Overhearing.* Due to the broadcast nature of wireless channels, many nodes in the vicinity of a sender node overhear its packet transmissions even if they are not the intended recipients of these transmissions [13]. This redundant reception results in unnecessary expenditure of battery energy of the recipients. Turning off neighboring radios during a certain point-to-point wireless transmission can mitigate this cost [13, 14].

*3.1.2. Received Signal Strength.* In wireless communication, received signal strength indicator (RSSI) has a crucial role in detecting received power because, it provides the information necessary to adjust the receiver's gain.

RSSI is the relative received signal strength in a wireless environment employing IEEE 802.11 expressed in arbitrary units. RSSI measurements range from 0 to 255 and are expressible as a one-byte unsigned integer. For typical wireless communication applications, RSSI circuits should have a wide range over 60 dB with fast settling time using the received start-up signal. The maximum value, RSSI_Max is vendor-dependent (i.e., Cisco Systems cards return a RSSI value from 0 to 100 where RSSI_Max is 100, and Atheros Wi-Fi chipsets return an RSSI value from 0 to 127 ($0 \times 7$f), with 128 ($0 \times 80$) indicating an invalid value).

Since RSSI is not absolutely accurate and stable, the performance of dissemination reliability estimation largely depends on estimation methods.

*3.2. Overview of the Proposed Approach.* Reliable data transfer using *each path reliability* in WSNs has not previously been discussed in the literature. The primary motive of LiReTa is to provide reliable end-to-end data delivery in a WSN environment with minimum energy expenditure. It uses an *each path-reliability approach* to select between $ACK_{dir}$ and $ACK_{imp}$. Further, LiReTa measures the path reliability using RSSI and channel error rate (CER), and also considers energy efficiency by minimizing the number of retransmissions and the number of ACK messages. In previous research, CER has only been applied to path reliability. However, because the operating condition of each node is influenced by the current network environment, the CER varies with time. Moreover, WSNs are composed of low energy power sensor nodes that are capable of sensing particular physical phenomena in their vicinities and communicating among themselves using wireless transceivers. Such low power wireless data communication features make WSN data dissemination unreliable. Therefore, we define reliability as the combination of CER with RSSI that represents the signal strength of neighbors for applying varying network conditions. RSSI is proportional to the input power level [15], and thus reliability using RSSI reflects the currently remaining sensor node power level, as shown in Figure 1.

We assume that sensor nodes are deployed in a general grid form in LiReTa. After sensor nodes are deployed, each sensor node shares RSSI values with neighboring nodes during network configuration. In this process, we calculate the path reliability using RSSI values and CER. When data is successfully transmitted, the calculated reliability is set as the threshold of reliability (used by base reliability later).

For data transfer, the node compares the base reliability and the current reliability. If the current reliability is lower than the base reliability, the $ACK_{dir}$ is requested. Otherwise transfer success is confirmed using $ACK_{imp}$, which is obtained because nodes used in wireless communications can overhear transmissions to other nodes (discussed in more detail in Section 3.7). Hence, when $ACK_{dir}$ is requested but the transmission success possibilities are higher in the next path, by applying $ACK_{dir}$ delegation to the next node, unnecessary traffic and overhead caused by ACK/NACK can be eliminated, which results in increased energy efficiency. This method detects errors between nodes to provide quick error recovery. Importantly, focusing on single path (single channel) reliability is a significant difference from previous approaches using combined channel reliability. The overall LiReTa procedure consists of 6 steps and is outlined briefly as shown in Figure 2.

*Step 1.* Sensor nodes are deployed in grid form (described in Section 3.4).

*Step 2.* Consider the number of retransmissions. If the network requires a limited number of retransmissions, consider the number of retransmissions in the ACK selection procedure (described in Section 3.5).

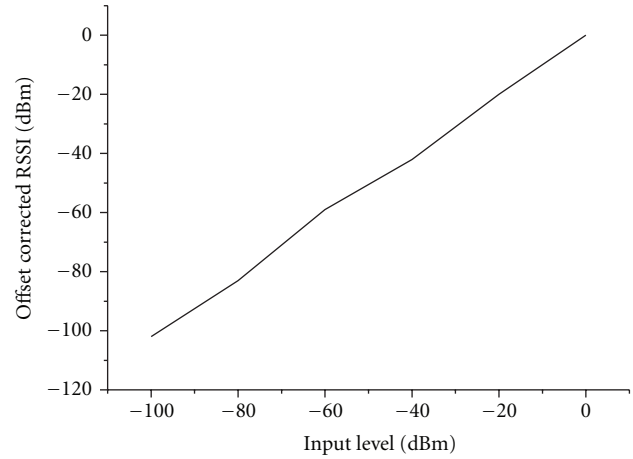



Figure 1: Offset corrected CC2520 RSSI versus input power level [15].

*Step 3.* Calculate reliability using RSSI and CER (described in Section 3.6).

*Step 4.* Select algorithm for reliable transmission considering overhead. That is, select $ACK_{dir}$ or $ACK_{imp}$ method according to path reliability considering the number of retransmissions (described in Sections 3.7.1 and 3.7.2).

*Step 5.* Delegate to improve energy efficiency. If $ACK_{dir}$ is used, then the node performs the $ACK_{dir}$ delegation process (described in Section 3.8).

*Step 6.* Update base reliability. If data transfer is successful, maintain the base reliability. If data transfer fails, increase the base reliability (described in Section 3.9).

In the proposed scheme, each node receives selective ACK feedback messages to guarantee reliability. The goal of LiReTa is to reduce the number of ACK messages using a selective ACK method that avoids the ACK implosion problem. In addition, well deployed sensor nodes such as grid topology can be used to reduce duplicated messages at each node, since a node can only send to the fixed neighbor node by using a well-coordinated node ID system. The $ACK_{imp}$ can be reduced when data communication channels are scheduled efficiently by a MAC layer.

Moreover, the $ACK_{dir}$ message is only used in the worst case when the path quality of a node is worse than the base reliability, or the number of network retransmissions is less than the required number of retransmissions. Use of the selective ACK method between $ACK_{imp}$ and $ACK_{dir}$ can thus reduce the ACK implosion problem significantly. In the next sections we explain each of these steps in more detail.

*3.3. Cross-layer Design.* Networks are organized as a series of layers, each one built upon the one below it. The main purpose of layered protocol architecture is to reduce

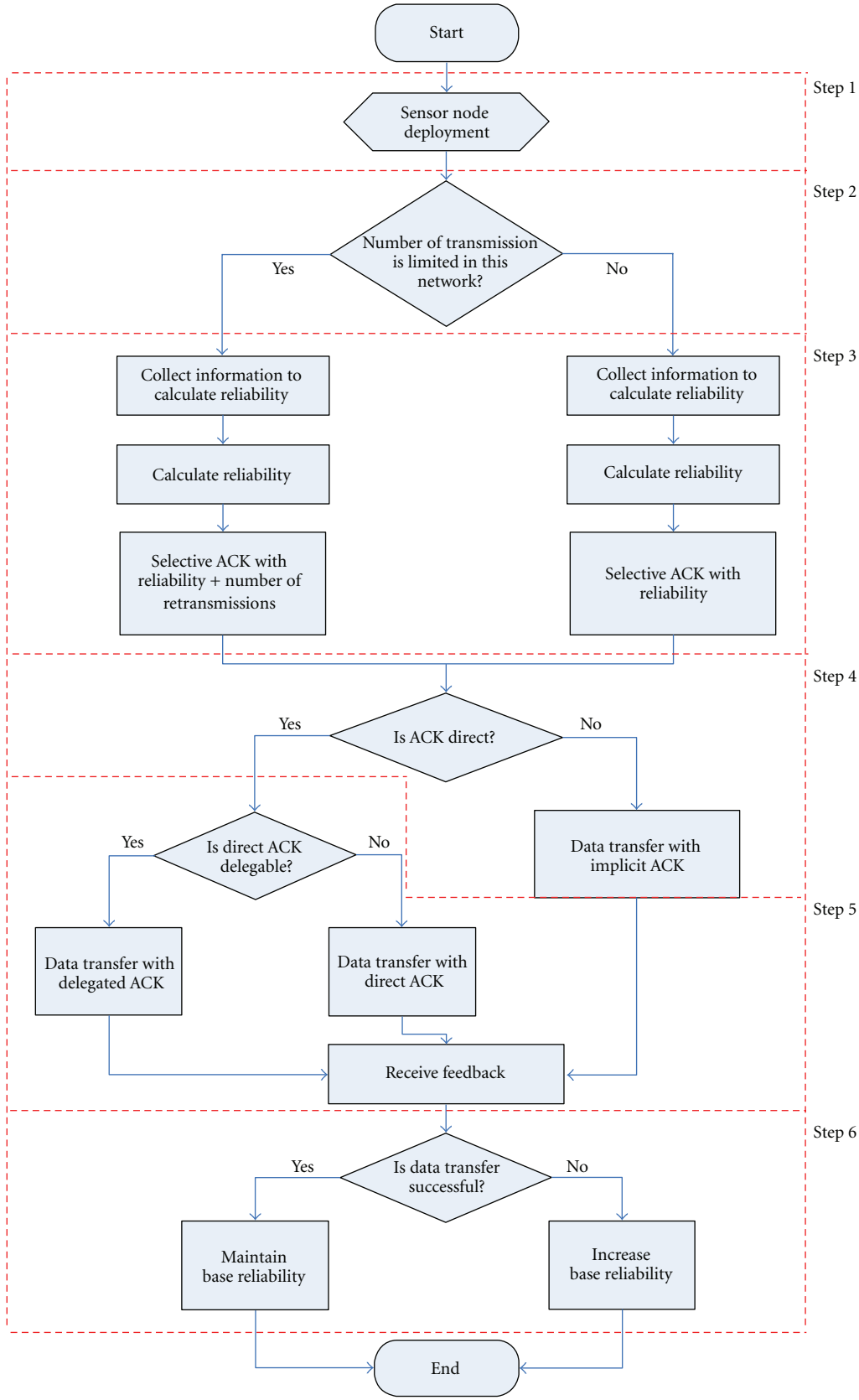



Figure 2: LiReTa procedure.

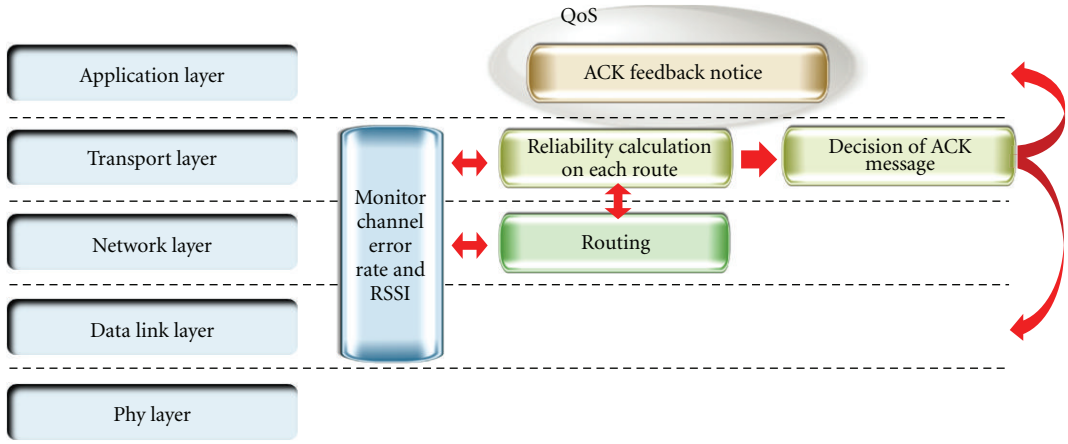



Figure 3: Cross-layer design in LiReTa.

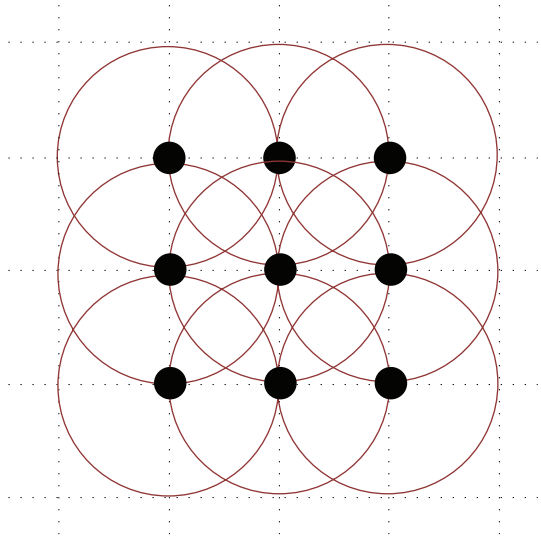


Figure 4: Sensor node deployment.

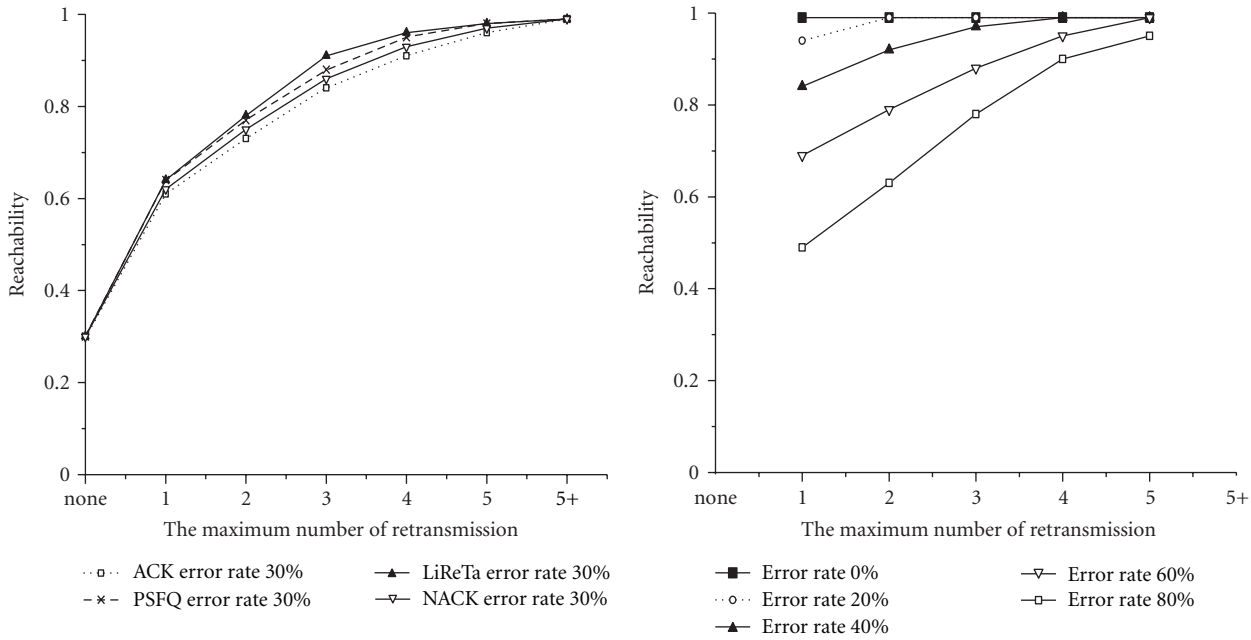
the complexity of system design. If the network is split into smaller modules with different functionalities, design is more manageable and implementation is easier. However, cross-layer design exploits the interactions between layers and promotes adaptability in all layers based on information exchange between layers. Moreover, cross-layer design produces tight interdependence between layers, especially in WSNs.

We designed LiReTa using cross-layer architecture with a holistic view of WSNs to maintain the layered approach, while accounting for interactions between various protocols at different layers. Figure 3 shows how cross-layer design is applied in LiReTa.

*3.4. Basic Node Deployment Structure of LiReTa.* The sensor node has mobility and works as both a host and a relay router. Therefore, broadcasting and multicasting are necessary for checking node position, signal strength, and network conditions [16, 17]. A *limited node transmission method* was studied to manage the excessive duplicated message problem known as a broadcast storm. These sets of message delivery nodes are connected dominating sets(CDS) within a given network and the solution to find the least-cost CDS was determined to be nonpolynomial complete (NP-complete) [18]. Thus, while various heuristics are used to find CDS, this paper will restrict CDS by deploying sensor nodes.

We assume that sensor nodes are deployed in a grid form as shown in Figure 4, which is a form that is commonly used in topology research; sensor nodes provide multipath routing to test neighbors' reliability and contains most of WSN constraints rather than other topologies. Most of the related work can also be easily implemented with this topology.

*3.5. Reliability and the Number of Retransmissions.* Our approach, LiReTa, also considers periodicity in WSNs since most applications in environmental, military, and medical environments sense and transmit data periodically. If there is no limit to the total number of retransmissions in WSNs,




(a) Transmission success rates with ACK, PSFQ, LiReTa, NACK

(b) Transmission success rates with 0%, 20%, 40%, 60%, 80% error rate cases

Figure 5: Transmission success rates with each scheme and error rate.

urgent data delivery is impossible due to surplus retransmission packets. Unlimited retransmissions also consume a great deal of energy. In general, the communication system limits the number of total retransmissions to prevent infinite transmission. Limited retransmission processes are key elements to support quality of service (QoS) for high performance networks. In WSNs with few sensor nodes, the number of retransmissions is critical for maintaining the reliability of whole networks. In this paper, we analyze the impact of retransmission for reliability as it applies to WSNs.

*3.5.1. Consideration of a Limited Number of Retransmissions.* We conducted several experiments to verify failure with a limited number of retransmissions. Simulations were performed on a single path topology with 10 nodes. Network conditions including packet loss rates and CER were randomly allocated within the range of error rates in the overall network. In our experiments, reachability was defined as the ratio of the number of packets that arrived at the sink node to the number of packets sent by a source node [19].

The reachability shown in Figure 5 converges to 1 when the number of retransmissions was limited to 5 by an NS-2 simulator, compared with ACK, PSFQ, NACK, and LiReTa schemes. In Figure 5(b), the results show varied reachability in terms of the number of retransmissions with different error rates. Error rates in the model were set at 0%, 20%, 40%, 60%, and 80%.

*3.5.2. Relationship between RSSI and CER.* As shown in Figure 6(b), when the number of retransmissions is 2 and the CER for two nodes is 80%, the success rate of packet delivery is very small. In order to find general metrics affecting retransmissions, RSSI and CER are investigated and analyzed as shown in Figure 6.

Based on Figure 6, the correlation analysis for RSSI and CER is calculated as follows:

$$\text{Correlation } P_{xy} = \frac{(1/n)\sum_{i=1}^{n}(x_i - \mu_x)\left(y_i - \mu_y\right)}{\sigma_x \times \sigma_y} = -0.9605, \tag{1}$$

where

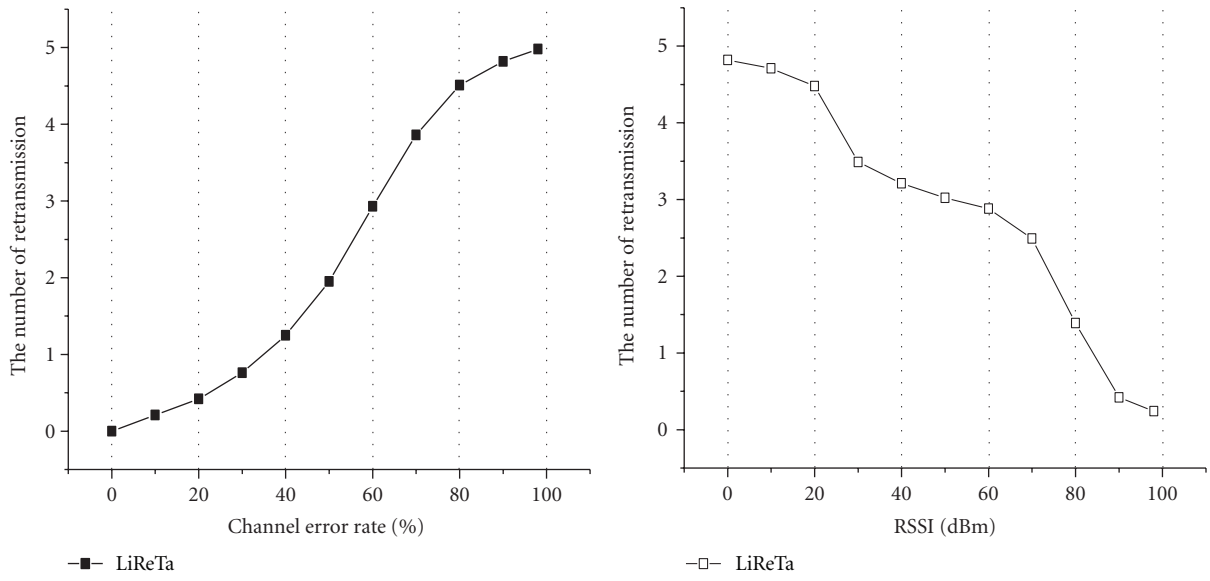
(i) $x$: average number of retransmissions of a sample for RSSI,

(ii) $y$: average number of retransmissions of a sample for CER,

(iii) $n$: sample size.

As shown in (1), the correlation between RSSI and CER with retransmissions is very high. We found that these two metrics are highly related to the number of retransmissions, and can be applied to design cross-layer-based reliable transmission.

*3.5.3. Appropriate Number of Retransmissions for CER and RSSI.* For mission critical systems requiring high communication environment reliability with fast delivery, setting a small number of retransmissions is appropriate. We simulate 100 packets data transmission between A-B nodes with 65% CER when the number of retransmission was limited 0 to 2 as shown in Table 3 by an NS-2 simulator. As tested, if the number of transmissions is set to 2, 20% failure occurs but an average of retransmissions observed by 1.84. To avoid




(a) CER with number of retransmission

(b) RSSI with number of retransmissions

Figure 6: RSSI and CER with Number of Retransmissions.

Table 3: Total retransmissions under the same conditions.

| Number of retransmissions (when data delivery succeeds) | Failure |
|---|---|
| 0 | 42 |
| 1 | 38 |
| 2 | 20 |

such failures in critical environments, configuring (setting) the number of retransmissions plays an important role in reliable transmissions.

To set the desired number of retransmissions $RT_i$ on node $i$ for successful delivery, we simply add the average and standard deviation of the number of retransmissions for successful delivery from our experiments to further increase the rate of successful delivery. Therefore, our approach is one in which each node decides to use $ACK_{dir}$ or $ACK_{imp}$ based on the path reliability and the required number of retransmissions, which previous approaches do not consider. $RT_i$ is calculated as follows:

$$RT_i = m + \sqrt{\frac{n\sum m^2 - (\sum m)^2}{n(n-1)} - 1}, \quad (2)$$

where

(i) $m$: average number of retransmissions on node $i$,

(ii) $n$: sample size.

To utilize this information, we need to investigate more facts (correlations) related to retransmissions and CER, and retransmissions and RSSI. Many simulations are performed using NS-2 to determine the minimum transmission time reported ($RT_i$) for successful delivery. The results are listed in Tables 4 and 5, respectively. The $RT_i$ with CER and RSSI considered together is analyzed as shown in Figure 7.

*3.6. Reliability Calculation.* After sensor nodes are deployed and share information, each node calculates the path reliability for the selective ACK procedure. As mentioned above, our approach utilizes CER and RSSI for the reliability calculation as shown in Section 3.5. The monitored CER accumulates errors which occur on all paths. Generally, CER is average value of node $A$ in WSN. In other words, CER is average value of whole path in node $A$. For example, if a node $A$ and other neighbor nodes $B$, $C$, $D$ are deployed as shown in Figure 8, the CER of node $A$ denoted by $E_A$ can be calculated according to (3). Applying CER in each path separately is one of our contributions in this paper:

$$E_A = \frac{\sum_{k=1}^{n} E_{ik}}{n}, \quad (3)$$

where

(i) $A$: current node to calculate reliability,

(ii) $n$: number of neighbor nodes,

(iii) $E_A$: CER of node$A$,

(iv) $E_{AK}$: $A \rightarrow k$ path CER.

In this paper, the overall reliability ($E_A$) of node $A$ is separated into paths $A$-$B$, $A$-$C$, and $A$-$D$. If we consider the average of the CER or the summation of the CER, then some unfair and unreliable events may occur. For example, if the CER is 0.2, 0.6, and 0.1 for paths $A$-$B$, $A$–$C$, and $A$–$D$, respectively, node $A$'s averaged CER value is 0.3. The average CER for node A seems acceptable even though one path ($A$-$D$) has very poor reliability. Instead of using the average,

Table 4: Number of retransmissions required for CER.

| Channel error rate (CER) | Average | Standard deviation | Average + standard deviation | Total number of transmissions | Number of retransmissions |
|---|---|---|---|---|---|
| 0–10 | 0.07 | 0.256 | 0.326 | 1 | 0 |
| 10–20 | 0.23 | 0.529 | 0.759 | 1 | 0 |
| 20–30 | 0.48 | 0.741 | 1.161 | 2 | 1 |
| 30–40 | 0.70 | 1.078 | 1.778 | 2 | 1 |
| 40–50 | 1.03 | 1.159 | 2.189 | 3 | 2 |
| 50–60 | 1.18 | 1.274 | 2.454 | 3 | 2 |
| 60–70 | 1.78 | 1.345 | 3.125 | 4 | 3 |
| 70–80 | 2.26 | 1.368 | 3.628 | 4 | 3 |
| 80–90 | 2.78 | 1.292 | 4.072 | 5 | 4 |
| 90–100 | 3.63 | 1.397 | 5.027 | 6 | 5 |

Table 5: Number of retransmissions required for RSSI.

| RSSI | Average | Standard deviation | Average + standard deviation | Total number of transmissions | Number of retransmissions |
|---|---|---|---|---|---|
| (−100)–(−90) | 4.05 | 1.009 | 5.059 | 6 | 5 |
| (−90)–(−80) | 3.07 | 1.297 | 4.367 | 5 | 4 |
| (−80)–(−70) | 2.67 | 1.477 | 4.147 | 5 | 4 |
| (−70)–(−60) | 2.28 | 1.505 | 3.785 | 4 | 3 |
| (−60)–(−50) | 1.85 | 1.566 | 3.416 | 4 | 3 |
| (−50)–(−40) | 1.65 | 1.572 | 3.222 | 4 | 3 |
| (−40)–(−30) | 1.01 | 1.275 | 2.285 | 3 | 2 |
| (−30)–(−20) | 0.46 | 0.797 | 1.257 | 2 | 1 |
| (−20)–(−10) | 0.25 | 0.575 | 0.825 | 1 | 0 |
| (−10)–(−0) | 0.08 | 0.273 | 0.353 | 1 | 0 |

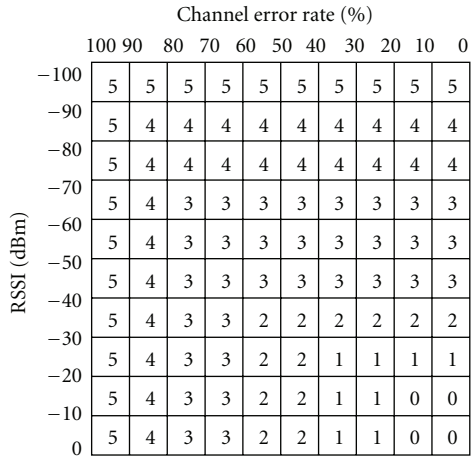



Figure 7: Retransmissions resulting in successful delivery with CER and RSSI considered together.

we need to use the CER of each path to differentiate the *reliability of each path*.

We previously discussed the very strong relationship between CER and RSSI in Section 3.5. The RSSI value of a node allows neighbors to recognize that the sender node is healthy simply by observing the received field in the L2 (data link layer) frame, which enables a node to determine the reliability of the neighbor nodes. RSSIs are basically negative forms; therefore, normalization steps are required as follows:

$$\mathrm{NRSSI}_{AK} = 1 - \left| \frac{\mathrm{RSSI}_{AK}}{\mathrm{MaxRSSI}} \right|, \tag{4}$$

where

(i) $\mathrm{RSSI}_{AK}$: $A \rightarrow K$ RSSI,

(ii) $\mathrm{NRSSI}_{AK}$: $A \rightarrow K$ normalized RSSI.

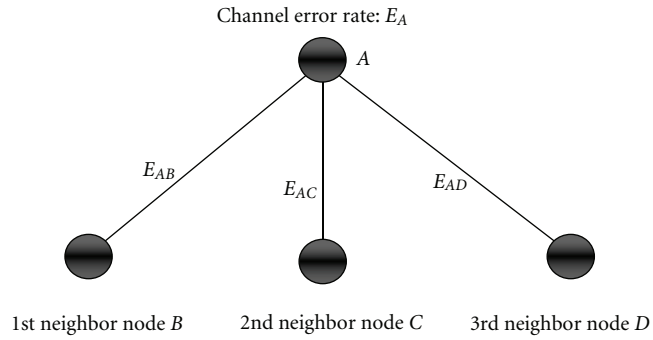



Figure 8: CER of node $A$.

From the average of CER of node $A$, using (3) and (4), we can calculate CER of specific path $E_{AK}$. In this paper, we present the reliability of each path. Hence, from the $1 - E_A$ equation, we can get the reliability of average of node $A$. For the separate path reliability, $R_{AK}$, the path reliability from node $A$ to $K$, is calculated as follows:

$$R_{AK} = (1 - E_A) \times \frac{QRSSI_{AK}}{\sum_{k=1}^{n} QRSSI_{Ak}}, \quad (5)$$

where $R_{AK}$: $A \rightarrow K$ is the path reliability.

### 3.7. Selective ACK Method

*3.7.1. Base Reliability.* When a node successfully exchanges an RSSI, it computes the reliability using the received RSSI to set a base reliability, that is, a threshold value for the trigger request of the ACK. If the transfer to exchange RSSI fails, the ACK request is sent until the node reaches the base reliability. The reliability calculation is calculated using (2) and (3), and used as the initial base reliability.

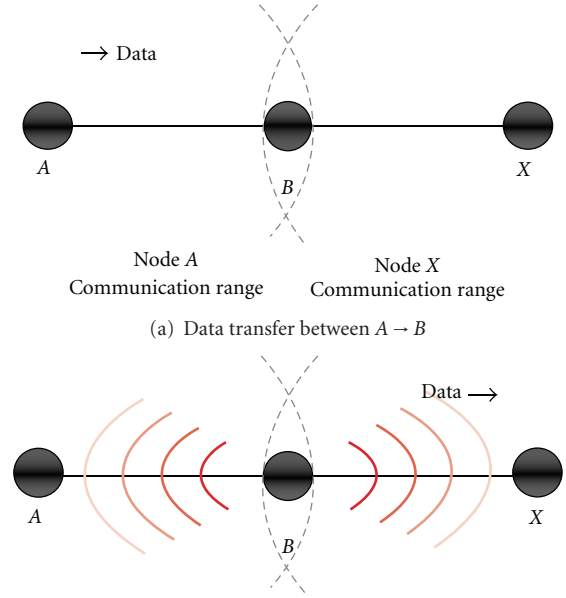
*3.7.2. Implicit ACK.* A sensor node uses radio channels for communication. Due to the broadcast nature of the wireless channel, many nodes in the vicinity of a sender node overhear its packet transmissions even if those are not the intended recipients of these transmissions [13]. This redundant reception results in so-called overhearing problems in IEEE 802.15.4 protocols. Turning off neighboring radios during point-to-point wireless transmission can mitigate this cost [13, 14]. Such overhearing problems are used positively as an implicit acknowledgement mechanism in the proposed LiReTa scheme.

Figure 9 shows the $ACK_{imp}$ mechanism. Using the selective ACK method, we determined whether current path reliability was better than the base reliability to confirm data transmission achievement using $ACK_{imp}$. If path quality was lower than base reliability, $ACK_{imp}$ transfer failure becomes high. In this low path quality case, $ACK_{dir}$ was used to guarantee reliability and fast recovery. Otherwise, as in Figure 9(b), by listening to node $B$'s forwarding signal, node $A$ receives an implicit ACK message.

As shown in Algorithm 1, through comparing the current path reliability, $R_{AB}$, and the base reliability, $BR_{AB}$ (initially received RSSI), and checking $RT_A$ (required retransmissions for successful delivery) and the configured retransmission limit, $NT_A$, we decide on either $ACK_{dir}$ or $ACK_{imp}$. After the proper ACK method is selected, data forwarding is started for the next hop.

The case, $R_{AB} > BR_{AB}$ and $RT_i \leq NT_i$, means that current reliability is better than the base reliability and current path quality (RSSI value) of $A$ requires less retransmission than the configured retransmission limit. That is, the current path is very reliable for delivering data successfully.

The case, $R_{AB} > BR_{AB}$ and $RT_i > NT_i$, means that current reliability is better than the base reliability and that the current path quality (RSSI value) of $A$ requires more retransmissions than the configured retransmission limit. That is, base reliability is very poor, and therefore not enough data exist to determine if the current reliability is good, or $NT_i$ is configured to be too small to have a very strict QoS requirement.




(a) Data transfer between $A \rightarrow B$

(b) Data transfer between $B \rightarrow X$

Figure 9: Implicit ACK using overhearing.

*3.8. Delegation.* In this section, we classify four scenarios to compare current path reliability and base reliability for delegation of ACK requests between $A$-$B$ and $B$-$X$.

*Scenario 1:* $R_{AB} > BR_{AB}$ && $R_{BX} > BR_{BX}$. In this scenario, the quality of all current paths is satisfied. Nodes $A$ and $B$ do not request $ACK_{dir}$ when node $B$ sends data to node $X$. Likewise, node $A$ overhears this data transmission and accepts this as an $ACK_{imp}$.

*Scenario 2:* $R_{AB} > BR_{AB}$ && $R_{BX} \leq BR_{BX}$. In the second scenario, node $A$-$B$ path quality is satisfied and does not require $ACK_{dir}$ messages while node $B$ requests an $ACK_{dir}$ message. When the transmission begins, node $A$ receives $ACK_{imp}$ and node $B$ receives $ACK_{dir}$ from node $X$.

*Scenario 3:* $R_{AB} \leq BR_{AB}$ && $R_{BX} > BR_{BX}$. In this scenario, current path $A$-$B$ reliability is worse than base reliability $A$-$B$. Node $A$ confirms transmission success through an $ACK_{dir}$ message. Node $B$ compares the next path $B$-$X$ with base reliability and the path $B$-$X$ quality is confirmed. There is no request for an $ACK_{dir}$ message to node $X$. Even if node $A$ requests an $ACK_{dir}$ message, $B$-$X$ path quality is good enough to make an ACK message unnecessary. Therefore, node $B$ delegates $ACK_{dir}$ to node $X$ and node $A$ overhears this information as an $ACK_{imp}$.

```
*compare path reliability and required network retransmission limitation*/
If (R_AB > BR_AB)
{
  if (RT_i ≤ NT_i)
    Forward Data (B, X, Data); // ACK_imp
  else // RT_i > NT_i
    Forward Data (B, X, Data, ACK); // ACK_dir
}
else // R_AB ≤ BR_AB
{
   if (RT_i ≤ NT_i)
    Forward Data (B, X, Data, ACK); // ACK_dir
  else // (RT_i ≤ NT_i)
    Forward Data (B, X, Data, ACK); // ACK_dir
}
```

Algorithm 1: Selective ACK algorithm.

```
/*Change base reliability*/
// use ACK_imp, good path quality
If (R_AB > BR_AB)
    if (B, Recv Data (A))
    {
        BR_AB = BR_AB; // success: maintain base reliability
        Clear Buffer (A); // success: clear A buffer
    }
    else
        BR_AB = R_AB; // fail: increase base reliability
// use ACK_dir, bad path quality
Else // R_AB ≤ BR_AB
    if (B, Recv Data (A))
    {
        BR_AB = R_AB; // success : decrease base reliability
        Clear Buffer (A); // success: clear A buffer
    }
    else
    BR_AB = BR_AB; // fail: maintain base reliability
```

Algorithm 2: Base reliability change algorithm.

*Scenario 4:* $R_{AB} \leq BR_{AB}$ && $R_{BX} \leq BR_{BX}$. In the last scenario, the quality of path *A*-*B*-*X* is poor. Node *B* sends an $ACK_{dir}$ message when *B* receives data from *A*. Node *X* sends an $ACK_{dir}$ message when *X* receives data from *B*. In this case, path quality is disqualified for $ACK_{imp}$, and thus an $ACK_{dir}$ message is highly recommended. When node *A* transfers data to *B* after deciding between $ACK_{dir}$ and $ACK_{imp}$ algorithms, node *B* initiates a delegation process as described in the third scenario. If A requests $ACK_{dir}$ because the path reliability is poor, *B* must decide whether to use ACK immediately or to delegate ACK. If the current path reliability is better than the base reliability, data dissemination will succeed even though there are no ACK requests. Node *B* then compares the base reliability to the next path reliability to decide whether to delegate. In this case, one node request $ACK_{dir}$ from a former node and, if the next path reliability is higher than the base reliability, the node delegates $ACK_{dir}$ to the next node.

*3.9. Update Base Reliability.* After $ACK_{dir}$ or $ACK_{imp}$ selection, the type of ACK is marked and data is sent. Based on the success or failure of data transfer, the base reliability is changed. This base reliability change reflects current network and node conditions. The base reliability updates the algorithm for node *A*-*B* as shown in Algorithm 2.

If data transfer succeeds with $ACK_{imp}$ (meaning good path quality), base reliability is maintained. Otherwise, we need to replace the base reliability with the current path reliability. Likewise, if data transfer succeeds with $ACK_{dir}$ (meaning poor network quality), we update the base reliability with the current path reliability. If data transfer fails, we maintain the base reliability.

## 4. Use Case

In LiReTa, the data transfer procedure chooses the type of ACK to guarantee reliability: $ACK_{dir}$ or $ACK_{imp}$. In addition,

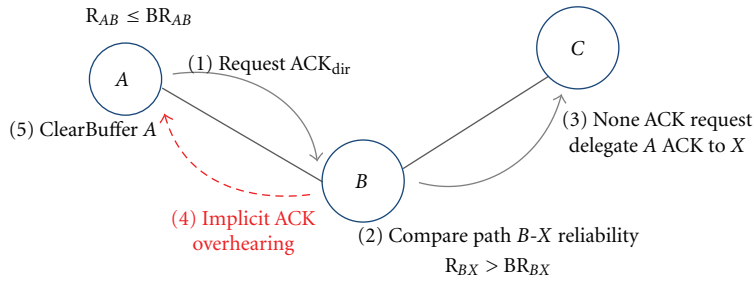



Figure 10: Delegation ACK.

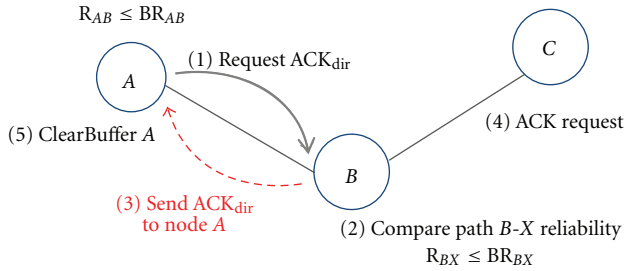



Figure 11: Receiving an $ACK_{dir}$.

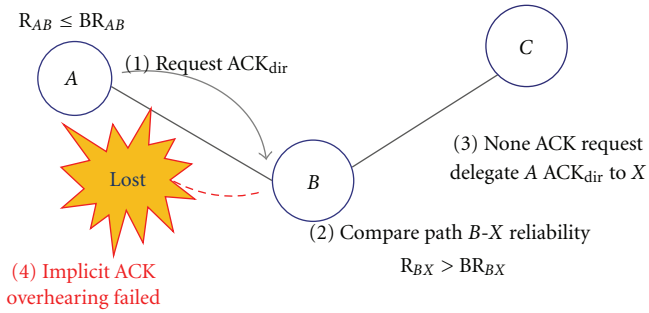



Figure 12: $ACK_{dir}$ failure.

based on data transfer success or failure, this process changes the base of reliability and determines the ACK delegation.

We can categorize all data transfer cases as follows:

(i) $A \longrightarrow B$ data transfer with $ACK_{dir}$, $B$-$X$ Quality is good enough to delegate direct Acknowledgment of node $B$ (denoted $ACK_{dir\_B}$) to $XB$-$X$ Quality is not good, $B$ returns $ACK_{dir\_B}$ to $A$ No response.

(ii) $A \longrightarrow B$ data transfer with $ACK_{imp}$, $B$ receives data successfully, and sends data to $X$ ($A$ overhears this transmission) No response.

In this section, we will explain these use cases in detail.

*4.1. A Data Transfer $A \longrightarrow B$ with an $ACK_{dir}$ Request:* $R_{AB} \leq BR_{AB}$

*4.1.1. ACK Delegation Occurs between Nodes B and X.* Node $B$ analyzes the chance of delegating after successful data transfer along the path $A$-$B$. When the path $B$-$X$ quality ($R_{BX}$) is higher than the path $B$-$X$ base reliability ($BR_{BX}$), $ACK_{dir\_A}$ is delegated to node $X$ even though the path $A$-$B$ quality ($R_{AB}$) is lower than the path $A$-$B$ base reliability ($BR_{AB}$) because the path $A$-$B$ data transfer has already succeeded. The path $A$-$B$ base reliability, $BR_{AB}$, must be decreased to the reliability $R_{AB}$ of the path $A$-$B$. This clears the buffer of $A$ as shown in Figure 10.

*4.1.2. Node A Receives $ACK_{dir\_B}$.* After $A$-$B$ data transfer success, node $B$ sends $ACK_{dir\_A}$ to node A when the path $B$-$X$ quality is lower than the $B$-$X$ base reliability. In the $R_{BX} \leq BR_{BX}$ case, node $B$ requests $ACK_{dir\_B}$ when node $B$ sends data to node $X$. At last, the buffer of $A$ is cleared due to $A$-$B$ data transfer success as shown in Figure 11.

*4.1.3. Waiting a Specified Time without Hearing from the Next Node.* This state may occur in two cases: data transfer failed from node $B$ or $ACK_{dir\_A}$ was delegated but node $A$ failed to overhear $ACK_{imp}$. The $ACK_{dir\_A}$ must be returned to node $A$ or node $A$ must overhear $ACK_{imp}$ since $ACK_{dir\_A}$ was requested in the first place. Since nothing was heard from node $B$, node $A$ sends data to $B$ with the $ACK_{dir\_A}$ as shown in Figure 12. The path $A$-$B$ current reliability decreases due to the incremental increase in the CER.

*4.2. Data Transfer $A \longrightarrow B$ with No $ACK_{dir}$ Request:* $R_{AB} > BR_{AB}$

*4.2.1. Node A Overhears Implicit Acknowledgment of Node B ($ACK_{imp\_B}$).* If node $A$ overhears data transfer between node $B$ and node $X$, data transfer from node $A$ to node $B$ succeeded. The buffer of $A$ is then cleared due to $A$-$B$ data transfer success as shown in Figure 13.

*4.2.2. Waiting a Certain Time without Hearing from the Next Node.* The data transfer from node $A$ to node $B$ failed as shown in Figure 14(a). In addition, Figure 14(b) shows that data transfer has succeeded but node $A$ failed to overhear $ACK_{imp}$. In this case, node $A$ sends data to node $B$ with $ACK_{dir}$ and increases $A$-$B$ base reliability $BR_{AB}$ to the current $A$-$B$ reliability $R_{AB}$. The current $A$-$B$ reliability decreases due to an incremental increase in the CER.

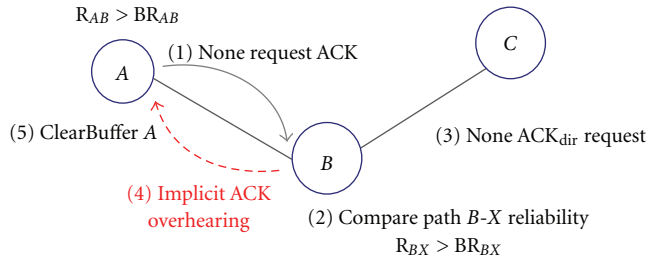



FIGURE 13: $ACK_{imp}$ overheard.

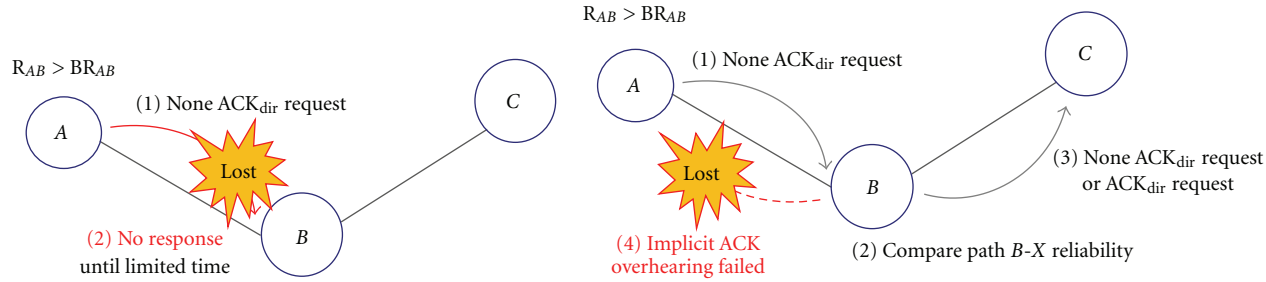



FIGURE 14: No ACK feedback.

## 5. Performance Evaluation and Analysis

*5.1. Simulation Environment.* In this section, we compare the performance results of LiReTa with generic results of ACK, PSFQ and ReInForM. The performance metrics include the number of transmissions, fault tolerance, node energy consumption, and network lifetime.

For the simulations, the sensor nodes transfer packets to the sink node on a particular routing path. The experiments for ACK and PSFQ methods assumed a fixed error rate. However, the proposed method considers variable reliability and thus only the initial error rate was set. The simulations considered the total number of transmissions including retransmission based on the number of nodes, time, and simulated error-recovery by increasing the size of packets. In addition, the proposed scheme compared energy consumption rates with the generic ACK method.

Simulations were executed over a uniform topology consisting of 0∼100 nodes deployed in a square grid form of 100 m × 100 m in the NS-2 network simulator. This method was selected due to the advantages of grid form as discussed in Section 3.4 with 1-hop neighbor overhearing communication radius of each sensor node. Network conditions were randomly allocated within the range of error rate parameter configurations. For example, if the range of the error rate was 50%, random rate values from 0% up to 50% were assigned to the packet loss rate and the CER, respectively. Each sensor node maintained a history of packet loss rates and a channel error count.

We applied the energy model, a network interface model, and an error model included in the NS-2 package. The initial energy level was set to 1 *J* and then compared to the generic ACK and pump-slowly fetch-quickly (PSFQ) protocols that are suitable for reliable transmission. To simplify the analysis, the traffic type was set as CBR, which generates packets periodically. The IEEE 802.15.4 package for NS-2 was used in the simulation. The maximum bandwidth was 250 kbps and the frames were transferred at a rate of 1 frame per second. The parameters used for energy consumption were the same as the sensor modes implementation using the CC2420 chipset [20] specification shown in Table 6. The parameter values were chosen considering the chip rate and bit rate of each radiofrequency band [11, 14].

*5.2. Results and Analysis.* For reliability simulation, we compared LiReTa with ACK method and PSFQ scheme. These two methods are the most relevant because of using ACK/NACK message and focused on reliable transmission. We choose these methods due to error recovery and the most well-known scheme in WSN. At first, we compare ACK method in view of confirmation for reliability. PSFQ helps to confirm LiReTa has good performance related to transmission time and total number of transmission.

For energy consumption simulation, we compared LiReTa with ACK method and ReInForM scheme. In related work, most of reliable transmission scheme focused on energy efficiency by reason of WSN nature.

*5.2.1. Reliability without Retransmission Limitation.* In this section, we present the results of transmission reliability simulations. We counted the number of transmissions until one packet successfully transmitted with no retransmission limitations in the network. The total number of transmissions, including retransmission, was used as a measurement of transmission reliability. The maximum number

Table 6: Parameters for power consumption analysis.

| Description | | 868 MHz | 915 MHz | 2.4 GHz |
|---|---|---|---|---|
| Power consumption | $P_{DOZE}$ | | 426 $\mu$A | |
| | $P_{Receive}$ | | 19.7 mA | |
| | $P_{Send}$ | | 17.4 mA | |

of transmissions was set to six times per packet (i.e., 5 retransmissions and 1 initial transmission).

The first simulation shows the total number of transmissions including retransmission for 3 protocols: generic ACK, PSFQ, and LiReTa based on a gradual increment in the number of nodes. Therefore, Figure 15 indicates the total number of transmissions. Including retransmission, when the initial CER was 30% and the number of nodes increased from 0 to 100. The hop-by-hop recovery method used by the PSFQ scheme showed the highest total number of transmissions when the number of nodes exceeded 50 hops while ACK and LiReTa transfer methods were relatively low. However, these three methods were similar within a 10% tolerance until 30 hops. Importantly, our proposed scheme maintained a maximum number of transmissions that was 5–10% lower than the ACK method via a reduction in unnecessary traffic as well as by applying path reliability data and utilizing confirmation messages when path quality was high.

In a simulation with no limitations on the number of retransmissions, the success rate of the proposed method was similar to that of ACK with respect to guaranteeing data transfer reliability.

In the second simulation, we used time to indicate the total number of transmissions. We assumed that the 10 nodes settled for 60 minutes. Thus, the adaptive nature of our algorithm was highlighted compared with other methods. Figure 16 shows similarly reliable performance for our algorithm, although the ACK mechanism exhibited more reliable performance than the PSFQ mechanism. Initially, PSFQ conducted quick error recovery; however, after 40 minutes, the LiReTa method resulted in a reduction in retransmissions of 25% compared to PSFQ and 7% compared to the generic ACK scheme. The buffer overflow problem occurred within the PSFQ scheme due to buffering for data storage in middle nodes until error recovery was completed. Thus, ACK and LiReTa yielded a decreased number of transmissions because these schemes do not accumulate data in buffers.

Moreover, using the same initial error rate, LiReTa showed slightly better performance than the generic ACK method.

Figure 17 shows the total number of transmissions with respect to increasing packet size. In this simulation, the ACK method did not consider packet size while PSFQ provided fast recovery. Initially, these three methods show similar result for the packet size. Therefore, the previous PSFQ simulations exhibited better performance than the ACK method and were similar to the proposed scheme. However, even though LiReTa showed an incremental increase in error rate as the packet length increased, it still had the best performance. Over than 100 bytes packets, PSFQ is increased

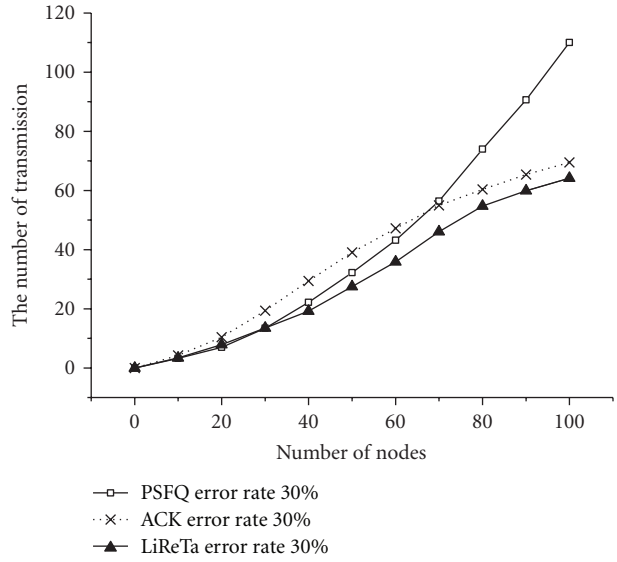



Figure 15: Total number of transmissions for an increasing number of nodes from 0 to 100.

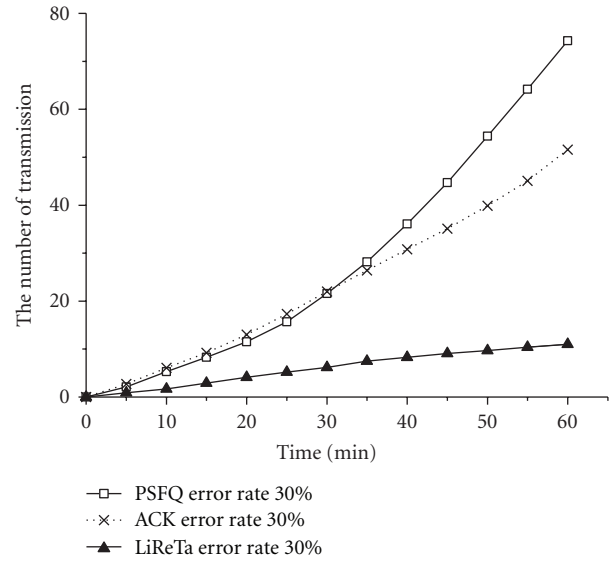



Figure 16: Cumulative transmissions for times between 0 and 60 min.

rapidly due to its recovery. 7% lower than ACK method and 24% lower than PSFQ when packet size is 150 bytes. LiReTa shows much better performance when packet size is bigger, for example, 11% more than ACK method and 38% more than PSFQ in 300 bytes packet.

Figure 18 shows the total number of transmissions with 5 case path qualities in Table 7. It is similar results in three methods when path quality is bad (case 1). In case 5, when path quality is good, LiReTa shows the best performance than 51.2% of ACK method and 37.5% of PSFQ in case 5. The main concept of LiReTa is not send ACK message when data transfer with high path quality. This simulation means that our proposed scheme still maintains the number of

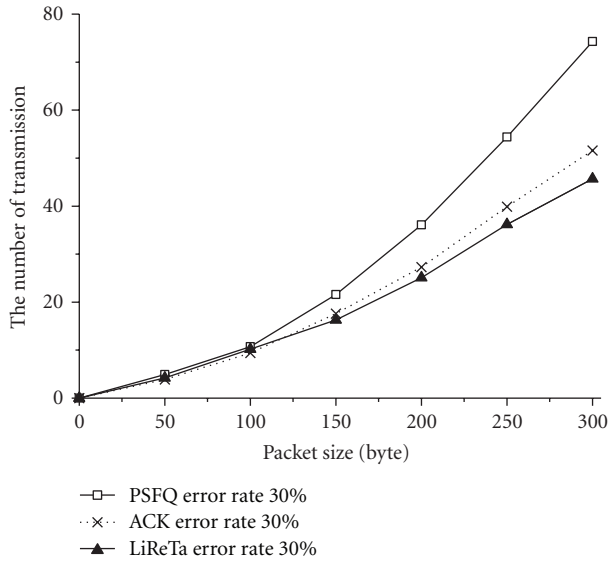



Figure 17: Total number of transmissions for packet sizes ranging from 0 to 300 bytes.

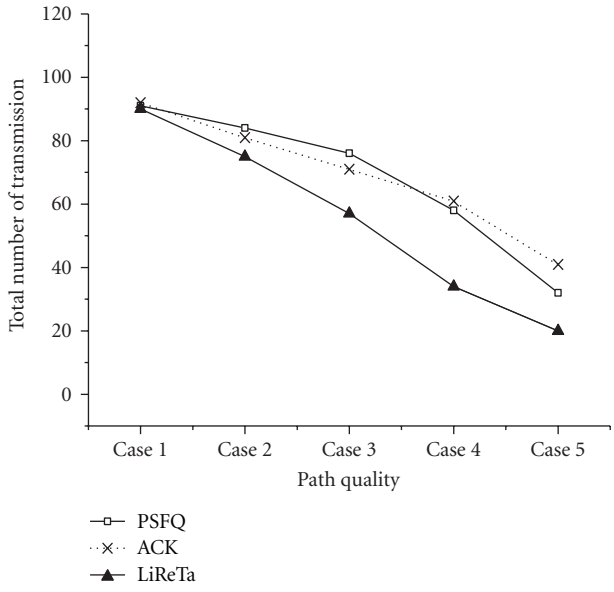



Figure 18: Total number of transmissions for path quality in 5 cases.

Table 7: Simulation conditions for path quality simulation.

| Case | Path quality |
|---|---|
| 1 | RSSI −75 CER 75 |
| 2 | RSSI −55 CER 55 |
| 3 | RSSI −45 CER 45 |
| 4 | RSSI −25 CER 25 |
| 5 | RSSI −5 CER 5 |

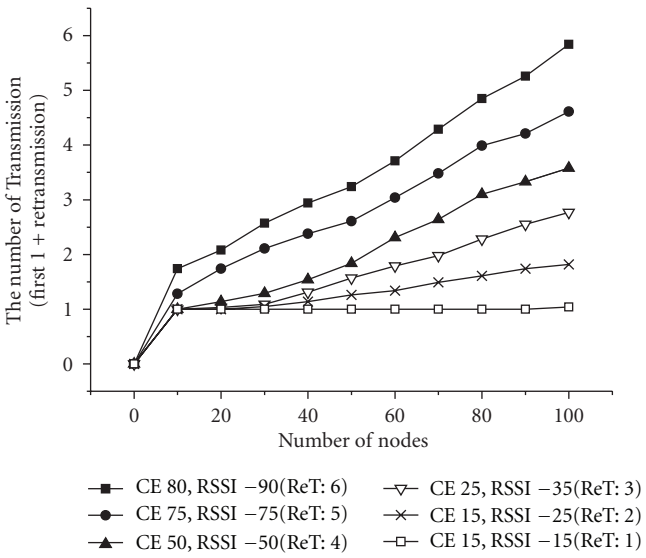



Figure 19: Average transmission time for each number of nodes for the six conditions in Table 8.

Table 8: Simulation conditions and number of minimum required transmissions for six simulations.

| No. | Simulation conditions | Number of minimum transmissions required $RT_i$ |
|---|---|---|
| 1 | CER 15, RSSI −15 | 1 |
| 2 | CER 15, RSSI −25 | 2 |
| 3 | CER 25, RSSI −35 | 3 |
| 4 | CER 50, RSSI −50 | 4 |
| 5 | CER 75, RSSI −75 | 5 |
| 6 | CER 80, RSSI −90 | 6 |

transmissions lower than ACK method and PSFQ reducing unnecessary traffic by applying path reliability and none confirmation message if path quality is high.

*5.2.2. Fault Tolerance with Limited Number of Retransmissions.* The goal of the simulation shown in Figure 19 was to examine the reliability of the required number of retransmissions for a period of time according to the node proposed using RSSI and CER in Section 3.5. Based on the power consumption needed for a given network time as defined in Table 6, we proceeded to validate our results from Section 3.5. Specifically, to identify data transfer success, the number of sent packets was compared to the number of received packets in a certain period of time.

The conditions of the simulation are defined in Table 8. The number of nodes ranged from 0 to 100 and data were transferred in a network with a number of nodes that increased at a rate of 10 nodes per trial, that is, 0, 10, 20 · · · 100. The maximum number of transmissions in each case was 100, as shown in Figure 19.

The simulated numbers of transmissions did not exceed the number of minimum transmission times required $RT_i$. Therefore, the number of transmissions required and the matching table calculation proposed in Section 3.5 (Figure 7) was shown to be valid within the proposed scheme.

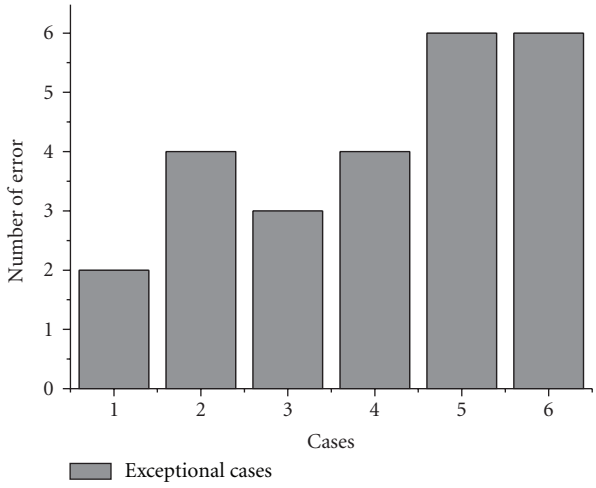



Figure 20: $RT_i$ < Transmission time for actual simulations of 6 conditions (Table 8).

Fault tolerances are shown in Figure 20. These unusual cases exceeded the number of minimum transmissions required compared to $RT_i$ values; however, the proportion was extremely low. Therefore, the average error rate was 4.1%, with a minimum error rate of 2% and a maximum error rate not exceeding 6%. This shows that reliability calculation with average plus standard deviation works well.

*5.2.3. Energy Consumption Comparisons.* To measure the energy consumption of each algorithm, we used the average consumed energy ($J$) of each node. The basic energy model in NS-2 applies different energy consumption levels except for other energy consuming stages such as routing. The initial energy was set to 1 $J$, idle electricity power to 5 $\mu$W, consumed power when receiving messages to 1.8 mW, and consumed power when sending messages to 27 mW.

*Total Energy Consumption Compared with no Limitations for the Number of Transmissions.* Figure 21 shows the energy efficiencies for error rates of 0%, 30, 60, and 98% when applied for ACK, ReInForM and LiReTa algorithms. The simulation results showed that the reliability of the proposed LiReTa scheme is as good as the other methods in terms of both reliability and energy efficiency. The ACK method energy consumption was set at 100% for the relative comparison between ACK, ReInForM and LiReTa. Our results showed that LiReTa had a 27% performance improvement while ReInForM had only a 20% performance improvement when the error rate was 0%. Moreover, when the error rate increased to 30%, the performance of LiReTa was improved by 29% while ReInForM had a 15% reduction in performance.

Therefore, when the CER was high, the number of retransmissions increased rapidly when using the generic ACK method. This was also the case for ReInForM due to the incremental increase in multi-paths needed to maintain reliability. On the other hand, LiReTa reduced the number of ACK messages by employing selective ACK and $ACK_{dir}$

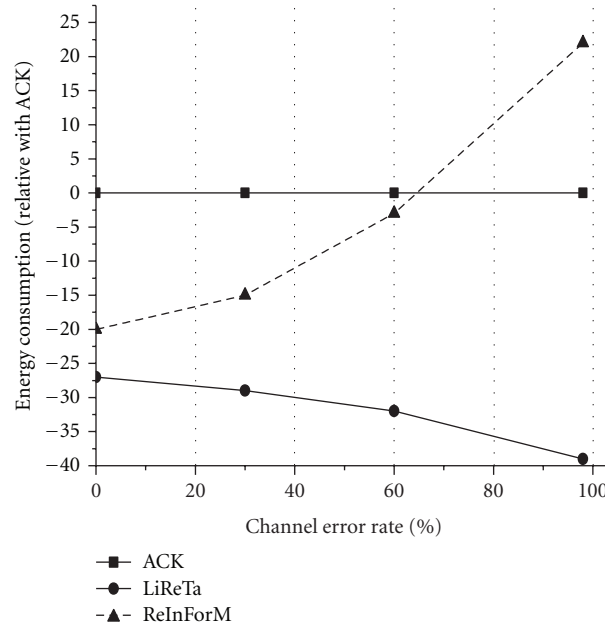



Figure 21: Energy consumption.

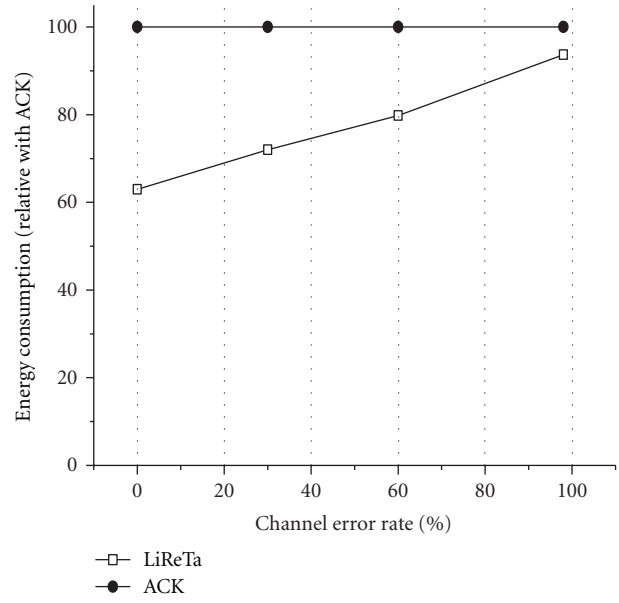



Figure 22: Energy consumption with a transmission limit of 4.

delegations. Hence, when the error rate was 60%, LiReTa had a 35% energy savings, which was even better than the ACK method, which had an average energy saving of 25%.

*Energy Consumption Comparisons with Number of Retransmission Limitations.* Moreover, we experimented with the initial error rate by setting it to 0%, 30%, 60%, and 98% in order to evaluate energy consumption when the transmission limit was initially set to 1 and the retransmission limit to 3 for a total of 4 transmission attempts. In this analysis, RSSI was set to −30% and the generic ACK method

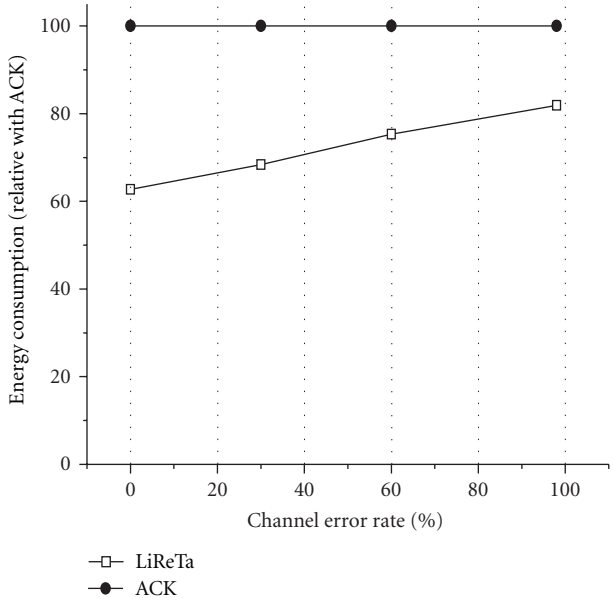



Figure 23: Energy consumption with a transmission limit of 6.

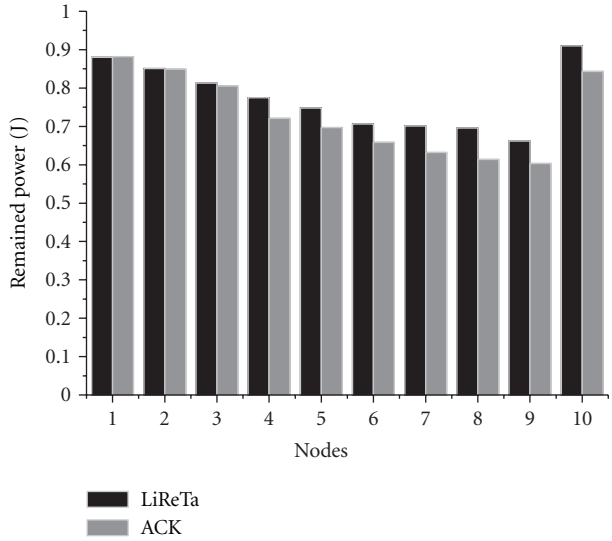



Figure 24: Remaining amount of energy in 10 nodes.

energy consumption was set to 100%. Figure 22 shows the proposed scheme employing overhearing and delegation with limited retransmissions. In this example, the energy consumption was reduced by up to 37% when the initial error rate is low. However, while the error rate increased, the ratio of $ACK_{dir}$ increased substantially, up to 100%. These results showed similar energy consumption rates among the different methods analyzed when the error rate was near 100%, which was due to $ACK_{dir}$ being used as many times as the generic ACK method. However, the LiReTa method showed more efficiency for larger retransmission limits as well as a lower initial error rate compared to the generic ACK method.

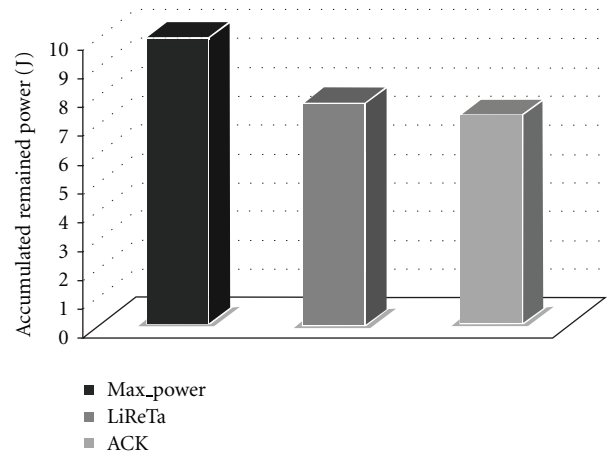



Figure 25: Total network lifetime.

Figure 23 shows the results of another experiment with the same conditions described in the previous experiment with the exception of a different total number of transmissions, 1 initial transmission and 5 retransmissions for a total of 6 transmissions. With 5 default retransmissions set in the MAC layer, the results of this experiment showed similar throughput to the case without retransmission limits. Further, the utilization of $ACK_{dir}$ gradually increased because most of the $ACK_{dir}$ were used while the number of transmission limits was replaceable with $ACK_{imp}$. This simulation showed that our scheme offers an average energy consumption rate that is 28% lower than the generic ACK method.

*Network Lifetime Comparison.* For network lifetime comparisons, experiments were performed over a linear topology comprised of 10 nodes and an initial CER set at 20%. We defined network lifetime as the total remaining power level in each node. Thus, we simulated the amount of energy remaining in nodes after 100 minutes, which we defined as network lifetime, as shown in Figure 24. The total amount of energy remaining is shown in Figure 25. The proposed scheme exhibited a 26% longer network lifetime compared with the generic ACK method.

Figure 26 shows two approaches for our proposal, namely, the LiReTa $ACK_{imp}$-based selective ACK method (IS-LiReTa) and the implicit ACK and delegation-based selective ACK method (IDS-LiReTa), where RSSI is fixed to −30 with variable error rates. The results show that IS-LiReTa and IDS-LiReTa had 22% and 30% higher energy efficiency compared with ACK, respectively. Finally, IDS-LiReTa, which utilizes delegation, exhibited a 35% performance improvement when the error rate was over 50%.

## 6. Conclusions

Since the power consumption of sensor nodes is highly influenced by data transmission, data loss must be minimized. Here, we present the LiReTa mechanism on WSNs for efficient data transmission. It considers network lifetime and

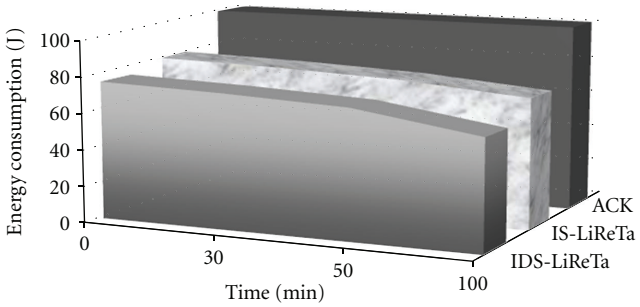



Figure 26: Energy consumption of generic ACK, IS-LiReTa, IDS-LiReTa protocols.

energy efficiency to support reliability for QoS requirements. To achieve reliable transfer in WSNs, several scenarios were discussed, implemented, and tested.

The contributions of this paper are threefold. First, we developed reliability calculations for each path. Second, we used the overhearing problem as an $ACK_{imp}$ for supplementary purposes when the path quality was good in order to avoid $ACK_{dir}$. Last, we developed a selective mechanism to delegate ACK mechanisms that reduces the ACK-implosion problem.

In addition, we extended a reliability calculation method that uses only channel error rate in the previous research to a process that employs RSSI values and calculates reliability for each individual node path. Energy and traffic waste were reduced. The buffer overflow caused by error recovery using middle node buffers, as shown in NACK and PSFQ [3], was also reduced by using the generic ACK method as a base.

The performance evaluations of this method employed the NS-2 tool to analyze reliable data transfer. Hence, the proposed scheme is efficient for reliable data transfer in WSNs and offers a new method to reduce overhead, thereby improving energy consumption. Indeed, the proposed scheme exhibits increased energy efficiency when the initial error rate is high.

In future work, we will consider presented LiReTa scheme in different network densities, random form of sensor nodes and large-scale scenarios with big data. Proposed method can be affected by any routing mechanisms; therefore, find a suitable routing algorithm can be one of key way to maximize the performance. In addition, we will gather more prerequisite elements to achieve accurate reliability and our research will focus on the impact of reliability in WSN data transfer.

## Acknowledgments

This work was supported by a Grant from the Kyung Hee University in 2011 (KHU-20111209) and the MKE (The Ministry of Knowledge Economy), Republic Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2012-(H0301-12-1004).

## References

[1] S.-J. Park, R. Vedantham, R. Sivakumar, and I. Akyildiz, "GARUDA: achieving effective reliability for downstream communication in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 2, pp. 214–230, 2008.

[2] H. Yang, T. S. Park, C. Y. Park, and C. I. Jung, "Selective unacknowledged transmission in IEEE 802.15.4 considering energy efficiency," *Journal of Information Scientists and Engineers*, vol. 16, no. 6, pp. 631–751, 2010.

[3] C. Y. Wan, T. A. Campbell, and L. Krishnamurthy, "Pump-slowly, fetch-quickly (PSFQ): a reliable transport protocol for sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 862–872, 2005.

[4] J. Rousselot, A. El-Hoiydi, and J. D. Decotignie, "Performance evaluation of the IEEE 802.15.4A UWB physical layer for body area networks," in *Proceedings of the 12th IEEE International Symposium on Computers and Communications (ISCC '07)*, pp. 969–974, 2007.

[5] B. Deb, S. Bhatnagar, and B. Nath, "Information assurance in sensor networks," in *Proceedings of the 2nd ACM International Workshop on Wireless Sensor Networks and Applications*, pp. 160–168, 2003.

[6] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: reliable information forwarding using multiple paths in sensor networks," in *Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks*, pp. 406–415, 2003.

[7] B. H. Lee, H. W. Yoon, J. H. Park, M. Y. Chung, and T. J. Lee, "Reliable transmission using intermediate relay node-based transmission for reliability, sensor network," *Journal of Korea Information*, vol. 30, no. 9, pp. 850–857, 2005.

[8] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz, "ESRT: event-to-sink reliable transport in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 5, pp. 1003–1016, 2005.

[9] C. Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: congestion detection and avoidance in sensor networks," in *Proceedings of the ACM SenSys Conference*, pp. 266–279, 2003.

[10] J. W. Choi and K. H. Lee, "A routing mechanism to prolong the lifetime of error-prone wireless sensor networks," *Journal of the Institute of Electronics Engineers of Korea*, vol. 46, no. 8, pp. 81–85, 2009.

[11] K. S. Yim, J. Kim, and K. Koh, "An energy-efficient routing and reporting scheme to exploit data similarities in wireless sensor networks," *Lecture Notes in Computer Science*, vol. 3207, pp. 515–527, 2004.

[12] S. J. Park and R. Sivakumar, "Congestion-aware topology controls for wireless multi-hop networks," *Ad Hoc Networks*, vol. 8, no. 3, pp. 295–312, 2010.

[13] K. I. Kim, "Sensor Network Technology-Analysis of Technology regarding energy efficiency issue," Report of KISTI Information Analysis team, 2008.

[14] Y. D. Yoo, J. H. Choi, and N. Kim, "Power consumption analysis of sensor node according to beacon signal interval in IEEE 802.15.4 wireless star sensor network," *Journal of Korea Information and Communications Society*, vol. 31, no. 9, pp. 811–820, 2006.

[15] P. Rand, "What's required for RF4CE, Resource document. The global electronics engineering community," 2008, http://www.eetasia.com/ART .

[16] Y. Tscha, "A reliable broadcasting scheme for directional antenna-based Ad Hoc networks," *Journal of Korean Institute of information Technology*, vol. 6, no. 6, pp. 66–74, 2008.

[17] Technologies and Savo G. Glisic, *Advanced Wireless Communications and Internet: Future Evolving*, John Wiley & Sons, 2011.

[18] H. Qayyum, S. Sohail, G. A. Shah, and A. Khanum, "A fuzzy decision maker for wireless sensor network in ubiquitous network environment," in *Proceedings of the International Conference on Advanced Information Networking and Applications Workshops*, pp. 359–364, 2009.

[19] J. W. Choi, B. Y. Choi, S. J. Song, and K. H. Lee, "NQAR: network quality aware routing in wireless sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, Article ID 409724, pp. 224–233, 2010.

[20] Texas Instruments, "Single-Chip 2.4 GHz IEEE 802.15.4 Compliant and ZigBee-Ready RF Transceiver, CC2420 Datasheet," 2009, http://focus.ti.com/docs/prod/folders/print/cc2420.html.